המדריך השלם

Access 2000
VBA

Microsoft Press

יש להתעלם מכל מה שנכתב על התקליטור
את קוד המקור ניתן להוריד מאתר הוד-עמי
בתיקיה "קבצי תרגול לספרים"

עורך ראשי ומקצועי: **יצחק עמיהוד**

תרגום: **במילים אחרות תיעוד בע"מ**

עריכה לשונית ועיצוב: **ענת קדם צבי, שרה עמיהוד**

עריכה מקצועית: **ארז ירון**

איל גוטליב, מנהל מרכז התמיכה של מיקרוסופט ישראל

אינדקס: **זהר עמיהוד**

עיצוב עטיפה: **סטודיו מצגר**

שמות מסחריים

הודעה

לשם שטף הקריאה כתוב ספר זה בלשון זכר בלבד. ספר זה מיועד לגברים ונשים כאחד ואין בכוונתנו להפלות או לפגוע בציבור המשתמשים/ות.

- טלפון: 09-9564716
- פקס: 09-9571582
- דואר אלקטרוני: info@hod-ami.co.il
- אתר באינטרנט: www.hod-ami.co.il

המדריך השלם

Access 2000
VBA

ריק דובסון

Designed for
Microsoft®
Windows NT®
Windows® 98

Microsoft Press

הוצאת הוד-עמי
לספרי מחשבים



Programming Microsoft Access 2000

By **Rick Dobson**

Published by **Microsoft Press**

Editor: **I. Amihud**

הוצאת הוד-עמי
לספרי מחשבים בע"מ
ת.ד. 6108 הרצליה 46160
טלפון: 09-9564716   פקס: 09-9571582
info@hod-ami.co.il

הודפס בישראל 2000
(כולל התאמה לעברית)

ISBN 965-361-206-9 מסת"ב

תוכן עניינים מקוצר

תוכן עניינים

פרק 2: מודלי גישה לנתונים 85

פרק 3: עיצוב טבלאות 155

פרק 4: טיפול בנתונים באמצעות שאילתות 199

פרק 5: בניית ממשק משתמש באמצעות טפסים............. 269

פרק 6: יצירת דוחות.. 301

פרק 7: מודולי מחלקה, טופס ודוח 335

פרק 8: אובייקטי Microsoft Office 369

הקדמה

על Access 2000

היישום Access הוא מערכת פיתוח פופולרית נפוצה ביותר, עקב היותו חלק מקבוצת יישומי Microsoft Office. לקוחות רבים מעוניינים שיישומי Access יפעלו בשיתוף עם שאר יישומי Office בצורה ברורה וקלה לתחזוקה, ללא צורך בתמיכת איש הפיתוח. הטכניקות המתוארות בספר יסייעו לך לענות על ציפיות אלו (לקבלת תמיכה נוספת, בקר בשני אתרי Microsoft: www.microsoft.com/office/ העוסק ביכולות הכלליות של Access, ו- www.microsoft.com/officedev/, הסוקר את רכיבי הפיתוח של המוצר).

רכיבים חדשים ב- Access 2000

Access 2000 מתקדם באופן משמעותי בתחומים רבים. Microsoft יצרה מוצר הכולל חידושים רבים, אך מתנהג עדיין כמו הגירסה המוכרת של Access. הספר מדגיש את החידושים בחמישה תחומים עיקריים:

- אובייקטי נתונים של ActiveX – ADO (ActiveX Data Objects),
- תפקודיות מורחבת של SQL Server בסביבות מחשוב שונות,
- Visual Basic for Applications (VBA) והרחבות אריזה (packaging enhancements),
- שיפורי מנגנון Jet,
- תפקודיות Web משופרת בסביבות מחשוב שונות.

אובייקטי נתונים של ActiveX

אובייקטי נתונים של ActiveX - ADO, מחליפים כמעט את כל פונקציות הגישה לנתונים שבוצעו באמצעות **אובייקטי גישה לנתונים** (Data Access Objects - DAO).
Access 2000 מציע תפקודיות ADO באמצעות שלוש ספריות : ADODB, ADOX ו-JRO.

הספריה **ActiveX Data Objects 2.1 (ADODB)** כוללת פונקציות עיבוד גישה לנתוני ליבה. האובייקטים העיקריים של ADODB הם : **Connection**, **Recordset** ו-**Command**. אפשר להשתמש באובייקטים אלה, במאפיינים ובשיטות (methods) שלהם, כדי להתחבר ולטפל במקור נתונים. לאובייקט Connection ממשק לטכנולוגיה חדשה של ספק OLE DB. לטכנולוגיה זו חשיבות חיונית עבור גישה אוניברסלית לנתונים – **UDA** (Microsoft Universal Data Access) המספקת גישה עתירת ביצועים למיגוון תבניות נתונים (יחסיות ולא יחסיות כאחד). טכנולוגיית UDA מאפשרת עיבוד משולב של מקורות נתונים מקובלים, כגון Jet ו- SQL Server, עם מקורות פחות מקובלים, כגון דואר, ספריות קבצים ואפילו וידאו. UDA מייצגת התקדמות התפתחותית מעבר לממשקי הנתונים המקובלים כיום, כגון קישוריות מסד נתונים פתוחה (Open DataBase Connectivity - ODBC), אובייקטי נתונים מרוחקים (Remote Data Objects - RDO) ו-DAO.

הספריה **Microsoft ADO Ext. 2.1 for DDL and Security (ADOX)** מציעה גישה מבוססת-אובייקט להגדרת נתונים ולאבטחה ברמת-משתמש. היא מספקת אוספי **משתמשים וקבוצות** (Users and Groups) ברמת משתמש מקובלת של Jet. היא גם קושרת הרשאות בקבצי מסד נתונים אל **חברים** (Members) באוספי **משתמשים וקבוצות** שבקובץ מידע של קבוצת עבודה (Workgroup information file). מודל ADOX של ספריה זו מטפל במטלות הגדרת נתונים באמצעות אובייקטים כגון **טבלאות** (Tables), **עמודות** (Columns), **אינדקסים** (Indexes), **מפתחות** (Keys), **תצוגות** (Views) **ושגרות** (Procedures). תוכל לנצל אובייקטים אלה כדי להגדיר בצורה דינמית טבלאות, אינדקסים וקשרי גומלין חדשים בין טבלאות. תוכל גם להגדיר שאילתות על טבלאות.

הספריה **Microsoft Jet and Replication Objects 2.1 (JRO)** מספקת בעיקר שירותי שכפול (Replication) של מנגנון Jet באמצעות ממשק ADO. המודל החדש ADO מאפשר לנצל את היתרון של רכיבי השכפול המתוכנתים של מסד הנתונים Jet. בנוסף, המודל כולל פונקציות של מנגנון Jet, כגון דחיסת מסדי נתונים ורענון המטמון.

הגברת הפעילות המשולבת של SQL Server

Access 2000 מאפשר לטפל במסדי נתונים מסוג Enterprise באותה קלות בה מטפלים במסדי נתונים מבוססי Jet, קישוריות ADO היא חלק מהסיבה לכך. סוג הקובץ החדש פרויקט Access (.adp), מאפשר גם השתלבות הדוקה עם גרסאות 6.5 ו-7 של SQL Server. סוג קובץ זה עובד עם מסדי נתונים מבוססי SQL Server ו- Microsoft Data Engine (MSDE) בדיוק כפי שקבצי mdb מאפשרים לטפל במסדי

נתונים מבוססי Jet. ‏MSDE הוא מנגנון מסד נתונים חדש שמבוסס על מודל SQL Server 7. הוא מיועד לפתרונות עבור קבוצות עבודה קטנות, ומשלים את תפקודי המנגנון המקובל Jet. תוכל לבחור לפתח פתרונות בעזרת Jet או MSDE.

הפרויקטים של Access מגלים תצוגות ושגרות במסגרת המוכרת של מסד הנתונים. תוכל להתחבר מיידית למסדי SQL מרוחקים תוך שימוש בממשק הגרפי הפשוט והמוכר ממסדי נתונים מסוג Jet. תוכל גם לנצל את נתוני SQL Server בעבודה עם טפסים ודוחות של Access (בדיוק כפי שאתה נוהג עם נתוני Jet).

לרשותך גם יכולת פעולה שקופה של OLE DB עם SQL Server ומקורות נתונים עורפיים (Back-End) בסביבות מחשוב שונות. השימוש באובייקט **Connection** של ADO ובספק OLE DB מאפשר להתחבר למקורות נתונים מרוחקים ולהפנות (reference) אליהם באמצעות קוד עבור יישומים מותאמים אישית.

VBA והרחבות האריזה

Access 2000 ממשיך בהתאמת גרסת VBA שלו ל-VBA של שאר יישומי Office. ‏Access 2000 כולל עורך – VBE ‏(Visual Basic Editor) שממשק המשתמש שלו זהה לזה של Word, ‏Excel ו-PowerPoint. ניתן להעביר את מיומנויות ניהול הקוד והפיתוח ישירות אל שאר היישומים, ועל ידי כך להעשיר אותם בפונקציות גישה לנתונים.

מהדורת הפיתוח של Office 2000 כוללת אפשרויות אריזה והפעלה משופרות. לדוגמה, ניתן להפעיל פתרונות באמצעות MSDE ופתרונות המתבססים על מסדי נתונים דמוי-SQL Server. סידור זה מאפשר לקבל את עושר התצוגות והשגרות של הממשק הגרפי וממשק התכנות. הדבר חשוב במיוחד במקרה של יישום קטן שמתפתח וזקוק מעתה ליכולות של מסד נתונים SQL Server.

ל- Access 2000 יכולת הטמעה חדשה המאפשרת להפיץ חבילות התקנה מותאמות אישית (custom setup packages) באמצעות האינטרנט. כך תוכל להרחיב במידה רבה את תחום לקוחותיך. מאגר הלקוחות הפוטנציאליים שלך יגדל ויקיף את כל מי שמחובר לאינטרנט, גם אם הוא נמצא בקצה העולם.

שיפורים במנגנון Jet

Access 2000 כולל את גירסה 4 של מנגנון מסד הנתונים Jet, המציעה שיפורים בתחומים תפקודיים אחדים. מצודדת במיוחד היא זמינות נעילת דף ברמת השורה (row-level page locking). הנעילה הנמוכה ביותר של גרסאות קודמות של Access היתה ברמת הדף. אחת הסיבות להכללת נעילה ברמת השורה היא זמינות תמיכת Unicode בתווי טקסט. ב- Office 2000 ניתן לייצג נתוני טקסט בשפות שונות ביישומים רב-לשוניים. שיטת הקידוד החדשה של שדות טקסט מגדילה את הזיכרון הדרוש לאחסון תו בודד מבית אחד לשני בתים, והדף גדל מ-2kb ל-4kb. מאז העליה בגודל הדף, Microsoft אפשרה לנעילה ברמת השורה לצמצם את אפשרות נעילת דף בו-זמנית ביישומים מרובי משתמשים.

גם בשכפול מסד נתונים חלו מספר שיפורים, אחד מהם הוא זמינות **שכפול ברמת העמודה** (column-level replication). גרסאות Jet קודמות זיהו התנגשויות ברמת השורה, שני עותקים התנגשו אפילו אם שינו שדות שונים באותה רשומה. שכפול ברמת העמודה משפר את הביצועים על ידי הסרת התנגשויות אלו. Access 2000 כולל גם שכפול דו-כיווני בין מסדי הנתונים Jet ו-SQL Server. הגירסה הקודמת אפשרה שכפול חד-כיווני בלבד, מ- SQL Server אל Jet.

עוד שיפור ראוי לציון הוא בקרה תכנותית שבאמצעותה משנים את ערך שדות המספור האוטומטי (AutoNumber). בעת יצירת טבלה תוכל להגדיר את הערך ההתחלתי ואת ערך הצעד (Step) של שדות מספור אוטומטי. תוכל גם לשנות ערכים אלה לרשומה הבאה. בפרויקט Access ניתן להגדיר שדות מספור אוטומטי מתצוגת **עיצוב טבלה** (Table Design). אפשר גם לשנות ערכים אלה לאחר היצירה הראשונית של הטבלה.

Jet אף מציע גישה ברמת SQL לתצוגות ושגרות. שיפורי SQL של Jet מאפשרים ליצור או לשנות את שני הסוגים של מודולי אובייקט מסד הנתונים.

הגברת הפעילות המשולבת ב-Web

אחת התכונות החדשות והמשמעותיות של Access 2000 הוא דפי גישה לנתונים, המתנהגים כמו טפסים ודוחות Access ב-Web. תוכל לעצב דפי Web הקשורים ישירות למקורות נתונים מסוג SQL Server או Jet. דפים הפועלים כטפסים מאפשרים למשתמש לערוך, להוסיף ולבטל רשומות בצורה גרפית מתוך הדף. תוכל להשתמש בכלי זמן-עיצוב (design-time) כדי לבקר באופן תכנותי תכונות אלו, וגם כדי לנצל את יכולות המיון והסינון. הדפים אינם מאפשרים יצירת טפסי-משנה, אך ניתן ליצור דפים מקובצים של גישה לנתונים (Grouped data access pages) שגדלים באופן מותנה בהתאם לקלט המשתמש.

דפי גישה לנתונים יכולים לשמש גם כמארחים של רכיבי אינטרנט חדשים, וניתן להשתמש בהם ליצירת דפים המכילים גליונות אינטראקטיביים, תרשימים דינמיים וטבלאות ציר (pivot tables). ניתן גם לקשור את רכיבי ה-Web של גליונות ותרשימים לנתונים המוצגים בדפים מקובצים או בלתי מקובצים של גישה לנתונים. משמעות הדבר היא שבאפשרותך להציג חישובים ותרשימים המשתנים בצורה דינמית בעת המעבר מרשומה אחת לבאה אחריה. דפי גישה לנתונים המכילים טבלאות ציר אינם מתקשרים עם מקורות נתונים אחרים בדף, אלא מספקים "סידור בטבלת ציר" (pivoting) בסגנון Excel, על ידי הנעה גרפית של חלקי נתונים בתצוגה נפרדת. בנוסף, ניתן לנצל טבלאות ציר לניתוח סוגי נתונים רבים, לרבות מקורות נתונים של SQL Server, Jet ועיבוד ניתוחי מקוון - OLAP (OnLine Analytical Processing).

על הספר

סגנון הלימוד של הספר מיועד להעניק לך יכולת מעשית כאיש פיתוח, ולהשיג תפוקה גבוהה בעבודה עם הרכיבים החדשים. הלימוד הוא בסגנון **הבט**, **ראה**, **פעל**! דוגמאות הקוד הרבות שתמצא בספר ממחישות את התפיסות שניתן לשלב ביישומים באופן מיידי. התייחס אליהן כאל מתכונים לביצוע מטלות פיתוח מוגדרות. נסה להפעילן "כמו שהן", ואחר כך התאם אותן לעבודה עם הנתונים שלך ובהקשר של היישום שיצרת. הדוגמאות פשוטות למדי, קלות להבנה ולשימוש חוזר, עובדות שתדרבנה אותך ליישמן.

קהל היעד

ספר זה מיועד למפתחי מסדי נתונים המתכוונים לפתח יישומים מותאמים אישית באמצעות Microsoft Access 2000. הספר דן ברכיבים ההופכים את Microsoft Access לכלי המועדף על מפתחים, ובחידושים המיוחדים של Access 2000. הספר מציג דוגמאות קוד רבות המדגימות את טכניקות הפיתוח העיקריות, וכלים לפיתוח יישומים מהיר. פשטות העיצוב נשמרה בקפדנות, כדי לאפשר לקורא לאמץ את קטעי הקוד שבספר ליצירת יישומים מותאמים אישית, או ליישום הטכניקות השונות במהלך עבודתו. ~~**התקליטור המצורף לספר** מכיל את כל דוגמאות הקוד שבספר~~.

הספר מיועד לספק מענה לצרכי מיגוון קוראים. אנשי פיתוח ותיקים ימצאו בו מידע חיוני על הרכיבים המתקדמים הכלולים ב- Access 2000. קהל יעד נוסף הוא אנשי פיתוח שעובדים בסביבה שונה מזו של Microsoft, כגון dBase או Paradox. אלה מתמצאים במושגי פיתוח, אך אינם יודעים בהכרח כיצד ליישם את המושגים בסביבת Access. קבוצה נוספת היא של משתמשים מקצוענים השואפים ללמוד לפתח פתרונות באופן עצמאי במינימום זמן.

תוכניות וקוד המקור שבספר

כל קטעי הקוד המופיעים בספר מסופקים בתקליטור המצורף. תוכל להשתמש בהם בשלמותם כדי לחסוך זמן, ולמנוע טעויות הקלדה במהלך העבודה.

~~התקליטור המצורף~~

בתקליטור המצורף תמצא את תיקיית קבצי הקוד הרלוונטיים לספר זה - תיקיה בשם **59237** הנמצאת בתיקיה **Books**. צור תיקיה בשם Programming Access בכונן C ואחר כך העתק אליה את תיקיות הפרקים שבתיקיה 59237 (קטעי הקוד הרלוונטיים לפרק מסוים, מופיעים בתיקיה ששמה כשם הפרק).

~~**להרחבה על תכולת התקליטור ואופן השימוש בו, פנה לקובץ ONCD שבתקליטור.**~~

מוסכמות בהן נעשה שימוש בספר זה

הספר משתמש במספר מוסכמות ואלמנטים מיוחדים כדי להדגיש נושאים מסוימים, ולהקל על הקריאה והשימוש. מומלץ להתבונן בהם בטרם תחל בקריאה.

הרשימה שלהלן מפרטת את המוסכמות בהן עושה הספר שימוש :

- מילים ומשפטים המדגישים את כוונת המחבר, כמו גם מונחים חדשים, מופיעים **באותיות מודגשות**.
- קטעי קוד ופקודות לדוגמה מופיעים בגופן Tahoma וברקע אפור.
- חלקים שונים בקטעי הקוד מופיעים מדי פעם *בגופן נטוי* כדי לציין שאלה חלקים אשר עליך להוסיף או לשנות (למשל פקודות או שמות משתנים). שיטה זו מאפשרת לך לראות מה עליך להוסיף או לשנות על פי ההקשר.
- בשל מגבלות הרוחב של עמודי הספר, נאלצנו במקרים מסוימים לשבור שורות קוד ארוכות במיוחד למספר חלקים. במקרים אלה, מופיע בסוף השורה השבורה סימן המשך שורה (_). בעת העתקת הקוד למחשב תוכל להשמיט את סימני ההמשך ולכתוב את הקוד במלואו בשורה בודדת או לחילופין - להעתיקו כפי שהוא מופיע בספר. Access יידע לפרש את הקוד בכל אחת מהצורות שתבחר.



הערה:

כל התמונות המופיעות בספר לקוחות מהגירסה האנגלית של Office 2000. בדיקות נכונות המידע בוצעו על הגירסה העברית, עם התייחסות לגירסה האנגלית בסוגריים.

הערה נוספת: כאשר תמצא התייחסות ל-Excel 8, הכוונה היא ל-Excel 97.

1

היכרות עם VBA

לפני יותר מעשור שנים הציע ביל גייטס שפת מאקרו אוניברסלית ליישומים שולחניים. שפת Visual Basic for Applications (VBA) היא הגשמת החלום ויותר מזה. VBA של Microsoft Access 2000 משותף לכל יישומי Office, ואפשר למצוא אותו גם במוצרי תוכנה של יצרנים נוספים. VBA עומד גם בכל כללי התחביר של שפת התכנות העצמאית Visual Basic. VBA מאפשר למפתחים להשתמש בשפת תכנות יחידה בעשרות הקשרים, כשכל שעליהם לעשות הוא ללמוד מודל אובייקט חדש. VBA הוא ה"דבק" של יישום Access : הוא מחזיק הכל יחד ומעניק ליישום צורה.

ל- Access 2000 ממשק חדש - עורך Visual Basic (VBE), המופיע ביישומי Office. אך בוודאי תמצא עצמך מוסיף קוד VBA לטפסי Access המוכרים, ולא לטפסי המשתמש שבשימוש יישומי Office אחרים. זיווג טכנולוגיות זה ייראה לך טבעי מאוד.

בפרק זה נציג את VBA כפי שהוא מופיע ב- Access 2000, ונסקור את עקרונותיו הבסיסיים בהקשר לפיתוח יישומים. כמו-כן נציג את חידושי VBA העיקריים ונדגים טכניקות תמיכה בטפסים באמצעות קוד. אומנם מדובר בסוגיית פיתוח מקובלת ב-Access, אך קיימים מספר שינויים באופן בו מממשים אותה באמצעות VBA.

הפרק ידון בשבעה היבטי VBA ב-Access, ויסתיים בתיאור קצר של פקודות מאקרו.

- **אוספים** (Collections), **אובייקטים** (Objects), **מאפיינים** (Properties), **שיטות** (Methods) **ואירועים** (Events),
- **שגרות** (Procedures) **ומודולים** (Modules),
- ממשק **VBE**,
- **Jet**, סוגי נתונים והצהרות,
- לוגיקה מותנית ומבני לולאה,
- פונקציות מוכללות,
- ניפוי ולכידת שגיאות.

אוספים, אובייקטים, מאפיינים, שיטות ואירועים

Access 2000 תומך ב-VBA, מה שמאפשר פיתוח מונחה-אובייקטים. בסעיפים הבאים נעסוק בפיתוח מונחה-אובייקטים בהקשר של VBA ו- Access 2000. המידע מיועד למשתמשים מתקדמים העוברים לפיתוח יישומים מתוכנת, ולמשתמשים ברמה בינונית המעוניינים לסקור את נושא התכנות המונחה-אובייקטים בעזרת VBA.

אוספים ואובייקטים

Access 2000 מהווה סביבת פיתוח מונחה-אובייקטים. חלון מסד הנתונים שלו מאפשר למשתמש לגשת לטבלאות, שאילתות, טפסים, דוחות, מודולים ופקודות מאקרו. VBA הופך את כל הרכיבים האלה לזמינים יחד עם מיגוון רחב של תבניות מתוכנתות, כגון **ערכות רשומות** (recordsets) ואובייקטים של **TableDef**. כדי להפיק מ-VBA את מירב התועלת ב-Access, עליך להבין את נושא האובייקטים ומושגים נוספים הקשורים לנושא זה.

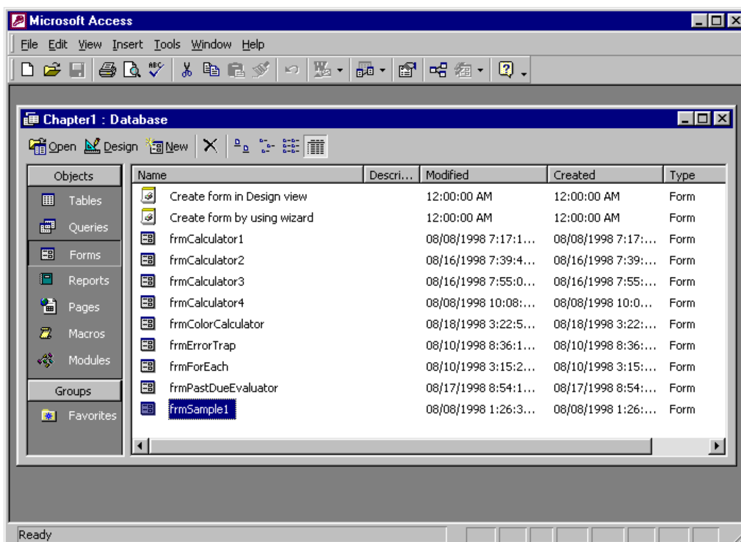
אובייקט (object) הוא עצם, כגון מכונית, טלפון או מכשיר וידאו. לכל האובייקטים יש מאפיינים. מכוניות למשל, מוגדרות באמצעות מאפייני הצבע, הדלתות, המנוע ומאפיינים נוספים. מאפיינים יכולים להגדיר מופעים של אובייקטים כלליים. התנהגות מונחית-אובייקטים מאפשרת להגדיר מופעים ייחודיים של אובייקטים בהתאם למאפייניהם. לדוגמה, מכונית אדומה ומכונית שחורה הן שני מופעים ייחודיים של האובייקט "מכונית".

מאפייני האובייקט משתנים בהתאם למחלקת האובייקט שאליה הם מתייחסים. למכונית יש קבוצת מאפיינים שונה מזו של מכשיר טלפון. שני האובייקטים מאופיינים באמצעות צבע, אך לטלפון יכול להיות גם מאפיין רמקול. למכוניות לעומת זאת, יש מנועים בנפחים שונים. יש אובייקטים שמהווים **מכולות** (containers) עבור אובייקטים אחרים. גם לאובייקטים המוכלים יש מאפיינים משלהם. מנועים מיוצרים במיגוון נפחים ותצורות, בעוד שלרמקולים יש בקרי עוצמה ומאפייני איכות צליל. מאפיינים יכולים להגדיר גם מופעים ייחודיים של מחלקות אובייקטים מוכלים. לטלפון עם רמקול (speaker) קבוצת מאפיינים שונה מזו של מכשיר טלפון רגיל.

בנוסף למאפיינים, אובייקטים כוללים גם שיטות. שיטות של אובייקט הן הפעולות שהוא מסוגל לבצע: מכשיר הטלפון משמש להתקשרות, מכונית נוסעת. אובייקטים רבים כוללים שיטות רבות. לדוגמה, מכשיר הטלפון מאפשר לבצע שיחות מקומיות ושיחות בין-עירוניות.

מפתחים בסביבת Access אינם מטפלים באובייקטים פיסיים, אלא בתבניות מתוכנתות, כגון טפסים, טבלאות ושאילתות שיכולים לייצג אובייקטים ופעולות שהם מבצעים. החלון **מסד נתונים** (Database) של Access 2000 מציג חלק ממחלקות האובייקטים של מסד הנתונים בסרגל הכלים המזכיר את זה של Outlook (ראה

תרשים 1.1). לחיצה על לחצן **טפסים** (Forms) שבסרגל הכלים פותחת תצוגה של אובייקטי טופס ומציגה שתי אפשרויות ליצירת טפסים חדשים. אובייקטי טופס מסוגלים להכיל אובייקטים נוספים שנקראים **פקדים** (Controls). אובייקטים מוכלים מגדירים אובייקט בדיוק כפי שפקדים בטופס מגדירים את מראהו והתנהגותו.



תרשים 1.1: החלון **מסד נתונים** ובו אוסף של אובייקטי טופס ושתי אפשרויות ליצירת טפסים חדשים (שים לב, התרשים לקוח מהגירסה האנגלית)

תרשים 1.1 מציג אובייקטי טופס אחדים. אובייקטים אלה מהווים **אוסף** (collection). באופן טיפוסי, יישומי Access כוללים אוספי טפסים, טבלאות, שאילתות ואובייקטים נוספים. החלון **מסד נתונים** עצמו ממיין אובייקטים לפי מחלקות. לחיצה על סרגל הכלים בחלק הימני (או השמאלי בגירסה האנגלית) מציגה בחלון את כל האובייקטים שבאוסף.

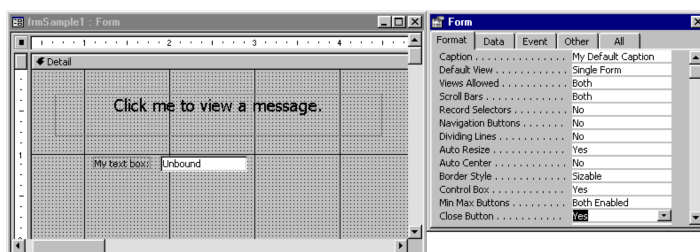
אוספים דומים מאוד לאובייקטים. לכל אוספי Access יש מאפיין Count (מונה) שמגדיר את מספר המופיעים באוסף. Item (פריט) הוא מאפיין נוסף של אוסף. תוכל לנצל את המאפיין **Item** לקריאה בלבד, כדי להחזיר טופס יחיד מתוך האוסף **AllForms** (כל הטפסים). **חברי** האוסף (members) הם אובייקטים יחידים, ולכן אין להם מאפיין Count. אובייקטי אוסף משמשים למטרות שונות. מכונית בעלת גג מתקפל יכולה למלא פונקציות שונות מאלו של מכונית משפחתית רגילה, אך שתיהן יכולות להשתייך לאוסף המכוניות של משפחה מסוימת.

מאפיינים ושיטות

המאפיינים והשיטות מאפיינים את מראה והתנהגות האובייקטים. תחביר ההפניה למאפיינים ושיטות הוא object.property ו-object.methods, בהתאמה. המונח object (אובייקט) יכול להתייחס לאובייקט יחיד או לאוסף אובייקטים. לדוגמה,

txtInput1.BackColor מציין את מאפיין צבע הרקע של תיבת טקסט בטופס, ואילו AllForms.Items(0) מתייחס לטופס הראשון באוסף הטפסים. אם שם הטופס הראשון frmSample1, ניתן להפנות אליו בתור AllForms.Item("frmSample1").

אפשר להציג את מאפייני אובייקט מסד הנתונים על ידי בחירתו בתצוגת **עיצוב** (Design) ולחיצה על לחצן **מאפיינים** (Properties) שבסרגל הכלים. תרשים 1.2 מציג טופס בתצוגת **עיצוב** ביחד עם גיליון המאפיינים שלו. גיליון המאפיינים מציג ערך מותאם אישית, My Default Caption במאפיין **כיתוב** (Caption). המאפיין **לחצן סגירה** (Close Button) נבחר. תוכל ללחוץ על שורת המאפיין **לחצן סגירה** ולבחור **לא** (No). שינוי הערך של המאפיין יציג באפור את הלחצן **סגירה** כאשר הטופס יופיע בתצוגת **טופס** (Form). שים לב שגיליון המאפיינים מכיל כמה דפים (כרטיסיות). תרשים 1.2 מציג את הכרטיסיה **תבנית** (Format) שבגיליון המאפיינים. הכרטיסיות מסדרות את המאפיינים בקבוצות, מה שמאפשר שליפה מהירה.



תרשים 1.2: טופס פשוט בתצוגת **עיצוב** יחד עם גיליון המאפיינים שלו



הערה:

ב- Access 2000 גיליון המאפיינים זמין גם בתצוגת **טופס**, ולא רק בתצוגת **עיצוב**. משמעות הדבר היא שניתן לשנות ולשפר את מראה הטופס בצורה קלה ופשוטה תוך כדי הצגתו בתצוגת **טופס**.

האובייקט DoCmd הוא מקור שיטות עשיר למפתחי יישומי Access בכל הרמות, אך מפתחים מתחילים ימצאו בו עזר רב בעבודה בסיסית עם שיטות. אובייקט זה כולל שיטות רבות, Close, OpenForm, GoToControl, FindRecord ו-RunCommand. שיטות רבות של אובייקט זה דורשות ארגומנטים המציינים את אופן הביצוע שלהן. שיטות אחרות כוללות ארגומנטים נדרשים או אופציונליים. אם אין מציינים ערכים של ארגומנט אופציונלי, השיטה מנצלת את הגדרות ברירת המחדל. השיטה RunCommand חביבה מאוד על משתמשים מתקדמים שעוברים לתכנות; ניתן לנצלה כדי לבצע את הפקודות הזמינות בתפריטים ובסרגלי הכלים של Access.

ב-Access אפשר לסגור טופס באמצעות השיטה Close של האובייקט DoCmd. לשיטה זו שני ארגומנטים נדרשים ואחד אופציונלי. הארגומנט הנדרש הראשון מציין את סוג האובייקט המיועד לסגירה. לסגירת טופס השתמש ב-acForm (acForm הוא קבוע מוכלל של Access שערכו מורה לשיטה Close כי ברצונך לסגור טופס. לקבלת מידע

נוסף על קבועים מוכללים ב-Access, עיין בסעיף "סורק האובייקטים"). הארגומנט השני הוא שם הטופס. ערך זה מופיע במאפיין **שם** (Name) בגיליון המאפיינים של הטופס. הצב את השם בין מרכאות. הארגומנט האופציונלי מורה ל-Access אם לשמור שינויים כלשהם בטופס. ברירת המחדל היא לבקש את אישור המשתמש לביצוע הפעולה. השתמש בשיטות acSaveYes או acSaveNo לסגירת הטופס תוך שמירת השינויים או ללא שמירתם. להלן תחביר המשתמש בשיטה Close לסגירת טופס:

```
DoCmd.Close acForm, "formname", acSaveNo
```

שיטות רבות של DoCmd חלות ישירות על אובייקטים יחידים. לדוגמה, השיטה GoToControl מעבירה את המיקוד לפקד מסוים בטופס. ניתן להשיג זאת גם באמצעות השיטה SetFocus שבוחרת את הפקד. נוח להפעיל את שתי השיטות כאשר על היישום להעביר את המיקוד לשם קליטת נתונים חדשים, או לתיקון מידע שגוי.

אירועים

אירועים הם מרכיבים חשובים מאוד בתכנות VBA. ניתן לנצל אותם כדי ליצור יישומים דינמיים ואינטראקטיביים. אירועים של אובייקטים ואוספים משמשים מעין נקודות הפעלה לקוד מותאם אישית שיצר איש הפיתוח. במהלך העבודה עם טפסים תוכל לנצל אירועים לביצוע מטלות, כגון אימות נתונים, הפיכת פקדים לזמינים או בלתי זמינים, העברת המיקוד לפקד אחר, ופתיחה וסגירה של טופס.

עליך להבין את תזמון האירועים וגם את סדר הפעלתם. פתיחת טופס מפעילה שורת אירועים: Open, Load, Resize ו-Current. האירוע Open מתרחש בעת שטופס מתחיל להיפתח, אך בטרם הוצגו רשומות כלשהן. האירוע Load מתרחש לאחר האירוע Open וגורם להצגת רשומות הטופס. קוד הגורם לטופס לשנות את גודלו או מיקומו באמצעות השיטות MoveSize, Minimize, Maximize או Restore של האובייקט DoCmd, מפעיל את האירוע Resize. האירוע Current הוא האחרון שמתרחש בדרך-כלל בעת פתיחת טופס. אירוע זה מסמן את הרגע בו רשומה מסוימת הופכת לרשומה נוכחית או לרשומה זמינה. האירוע מופעל גם כשהמשתמש מנווט לרשומה חדשה, מפעיל שאילתה, או מרענן טופס.

ניגשים לאירועי טופס על ידי בחירת הטופס או הפקד בתצוגה **עיצוב** ואחר כך בוחרים בכרטיסיה **אירוע** (Event) שבגיליון המאפיינים. לחיצה על הלחצן **בניה** (Build) שבשורת כל אחד ממאפייני אירוע, גורמת לפתיחת תיבת דו-שיח המאפשרת לפתוח את מודול הקוד שמאחורי הטופס. בחירה ב**בונה קוד** (Code Builder) גורמת לפתיחת שגרת אירוע ב-VBE. שגרת האירוע מקבלת את השם Objectname_Eventname, כאשר Objectname הוא שם האובייקט ו-Eventname הוא שם האירוע. למשל, אם בוחרים טופס ולוחצים על לחצן **בניה** להפעלת האירוע Close (בעת סגירה), שגרת האירוע תקבל את השם Form_Close. אם יוצרים שגרת אירוע עבור האירוע OnClick (בעת לחיצה) של תווית הנקראת lbTitle, היא תקבל אוטומטית את השם lbTitle_Click.

לפניך שלוש שגרות אירוע המתייחסות לטופס המוצג בתרשים 1.2: Form_Open, Form_Load ו-lbTitle_Click. בעת פתיחת הטופס בפעם הראשונה בתצוגת **טופס**, מופיעה תיבת הודעה ובה הודעה "The form opened" (הטופס נפתח). אישור ההודעה גורם להצגת הודעה נוספת, "The form loaded" (הטופס טעון). בתום טעינת הטופס, לחיצה על התווית גורמת לפתיחת תיבת הודעה שלישית, "Hello from the label".

```
Private Sub Form_Open(Cancel As Integer)
    MsgBox "The form opened.", vbInformation, _
        "Programming Microsoft Access 2000"
End Sub
Private Sub Form_Load()
' This is a simple statement.
    MsgBox "The form loaded.", vbInformation, _
        "Programming Microsoft Access 2000"
' This sets a property.
    Me.Caption = "New Caption"
' Here are two methods for giving a control focus.
'    DoCmd.GoToControl "txtMyTextBox"
     Me.txtMyTextBox.SetFocus
' Now that the method worked, VBA sets a property.
     Me.txtMyTextBox.Text = "Hi, there!"
End Sub
Private Sub lblTitle_Click()
      MsgBox "Hello from the label.", vbInformation, _
        "Programming Microsoft Access 2000"
End Sub
```

שגרות האירוע גורמות להצגת תיבות ההודעה. לחיצה על תווית מפעילה את שגרת האירוע lbTitle_Click. בשיגרה זו משפט יחיד שמציג את תיבת ההודעה (הקו התחתון בסוף השורה הראשונה של המשפט, מציין שהפקודה ממשיכה בשורה הבאה). שגרת האירוע Form_Open כוללת אף היא משפט אחד. שגרת האירוע Form_Load כוללת משפטים אחדים נוסף לזה שמציג את תיבת ההודעה. שגרת אירוע זו מגדירה בצורה דינמית את כיתוב הטופס, דבר זה שימושי במיוחד לטופס שממלא שני תפקידים או יותר באותו יישום. השיגרה גם מעבירה את המיקוד לתיבת הטקסט שנקראת txtMyTextBox ואחר כך מציבה את המחרוזת "Hi, there!" במאפיין Text של הפקד. שגרת אירוע זו מדגימה שתי טכניקות נפרדות לקביעת מיקוד. הראשונה מבוססת על השיטה SetFocus והשנייה על השיטה GoToControl. הגרש בתחילת שורות מסוימות מציין שהשורות הן הערות המיועדות לאנשים הקוראים את הקוד ו-Access אינו מתייחס אליהן. שים לב כי אחת הטכניקות מופיעה בשורת הערה.

שגרות ומודולים

שגרות הן מכולות (containers) של קוד VBA. קיימים שלושה סוגי מכולות: שגרות משנה (subprocedures), שגרות פונקציה (function procedures) ושגרות מאפיין (property procedures). למרות החפיפה במספר תפקודים, לכל שיגרה מטרה מוגדרת וייחודית לה בלבד. Access כולל שני סוגי מכולות בסיסיים: מודולים סטנדרטיים ומודולי מחלקות. מודולי מחלקות יכולים להיות מחלקות מותאמות אישית עבור טפסים ודוחות. תוכל לנצל אותם כדי להגדיר בעצמך מחלקות, ולפשט את השימוש החוזר בקוד של מטלות שגרתיות כגון הוספת עובד חדש, ביצוע הפקדה לחשבון או משיכה ממנו.

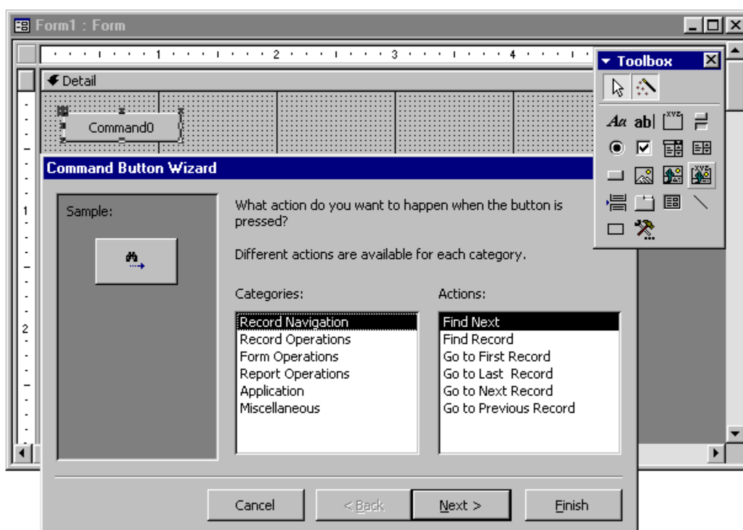
שגרות משנה

שגרות משנה יכולות לבצע פעולות, לחשב ערכים, לעדכן ולשנות הגדרות מאפיין מוכללות (לעיתים שגרת משנה נקראת בקיצור "שיגרה"). כפי שלמדנו, Access 2000 מפעיל אוטומטית שגרות כאשר מתרחשים אירועים, אך תוכל להרחיב את השימוש בשגרות אלו. שגרות לעולם אינן מחזירות ערכים וגם אינן מגדירות מאפיינים מותאמים אישית של טופס, דוח או מודול מחלקה.

שיגרה מכילה סדרת משפטי VBA בין המשפטים **Sub** ו- **End Sub** התוחמים אותה. המשפט Sub חייב להצהיר על שם שיגרה. שגרות אירוע מקבלות שמות מסוגננים מאוד (כגון object_event), אך שמות של שגרות נקבעים לפי כללי מתן שמות למשתנים. עליהם להתחיל באות, אורכם אינו יכול לעלות על 255 תווים, אין לכלול בתוכם סימני פיסוק או רווחים וגם לא מילות מפתח, שמות פונקציות או שמות אופרטורים של VBA. שגרות יכולות לקבל ארגומנטים הבאים לאחר שם השיגרה. אם השיגרה כוללת יותר מארגומנט אחד, יש להפריד את הארגומנטים בפסיקים.

אחת הדרכים להכיר טוב יותר את נושא השגרות היא השימוש ב**אשף לחצני פקודות** (Command Button wizard), אשר כותב קוד VBA למעל 30 פונקציות. כל שעליך לעשות הוא לבחור הגדרות חדשות בתיבות הדו-שיח. האשף מחבר שגרות פשוטות למדי, ולכן הן מהוות מקור יעיל ללימוד הנושא. גם מפתחים ברמה בינונית ומפתחים מתקדמים יצאו נשכרים משימוש באשף, מכיון שהקוד שהוא יוצר יכול לשמש כשלד בסיסי שעליו אפשר להוסיף קוד מפורט יותר. מפתחים מתחילים יכולים לנצל את האשף כדי להפוך תהליכים לאוטומטיים, כמו ניווט בין רשומות, תחזוקת מסד נתונים, טיפול כללי בטופס ובדוח, הפעלת יישומים נוספים ומטלות שונות נוספות, כגון הפעלת שאילתה או חיוג מספר טלפון.

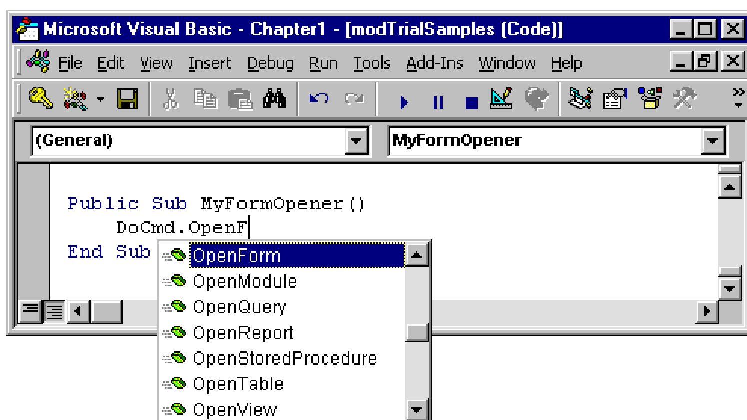
את האשף מפעילים מארגז הכלים שבתצוגת **עיצוב טופס**. בחר בלחצן **אשפי בקרה** (Control Wizards) ואחר כך בחר לחצן וצייר את הפקד על הטופס. תיפתח תיבת הדו-שיח המוצגת בתרשים 1.3. בכל קטגוריה מוצעות מספר פעולות. לאחר שתסיים את המעבר על תיבות הדו-שיח, תוכל להציג את הקוד ב-VBE. לחץ על לחצן **קוד** (Code) שבסרגל הכלים **עיצוב טופס** (Form Design) כדי לעבור אליו.



תרשים 1.3: אשף לחצני פקודות מאפשר ליצור שגרת אירוע מלאה תוך שימוש בתיבות דו-שיח אחדות

מפתחים רבים מעדיפים לכתוב שגרות מהתחלה (מבלי להתבסס על שיגרה קיימת). תוכל לפתוח את חלון הקוד וליצור שגרת אירוע כפי שתואר לעיל, או ליצור שיגרה רגילה. קיימות שתי דרכים להתחיל שיגרה כזו, זאת בהתאם למקום בו תציב אותה. אם השיגרה מיועדת לטופס או לדוח, לחץ על הלחצן **קוד** (Code) שבסרגל הכלים **עיצוב**. אם הקוד יופיע במודול סטנדרטי שאיננו מיועד לטופס או לדוח מסוימים, בחר **כלים** (Tools), **מאקרו** (Macro), **Visual Basic Editor**, או לחילופין הקש **Alt+F11**. חלון VBE ייפתח. בחר **Insert** (הוספה), **Procedure** (שיגרה), הקלד שם בתיבת הדו-שיח **Add Procedure** ואשר את בחירת ברירת המחדל של לחצן האפשרות **Sub**. כך תיווצר מעטפת לשיגרה, ובה המשפטים Sub ו- End Sub. כעת תוכל להוסיף לשיגרה קוד.

השתמש בידע שרכשת על מודל האובייקט של Access וכתוב מספר פעולות. זכור כי האובייקט DoCmd כולל שיטות רבות. הקלד DoCmd ואחר כך הקלד נקודה. תופיע תיבת רשימה נפתחת ובה כל הערכים החוקיים שיכולים להופיע לאחר DoCmd. במהלך הקלדת הערך, תוצג רשימת ערכים אפשריים התואמים להקלדה (תרשים 1.4). אם אינך בטוח בערך שעליך להקליד, גלול את רשימת הערכים ואתר את השיטה המבוקשת. טכניקה זו מתאימה לכל האובייקטים ולא רק ל- DoCmd. Microsoft קוראת לתכונה זו **IntelliSense**, מכיון שהיא מזהה בצורה נבונה את התשובות האפשריות. התכונה IntelliSense מבצעת למעשה שתי פעולות: מציגה את השיטות והמאפיינים החוקיים בכל שלב של הרכבת משפט VBA, ומספקת נתוני תחביר על תכולת השדות הדרושים למשפטי VBA שבבנייה. פעולות אלו מצמצמות באופן ניכר את שגיאות התחביר בקוד, ומסייעות לך להתחיל במהירות.



תרשים 1.4: התכונה IntelliSense מסייעת להשלים משפט VBA באובייקט DoCmd. בהקלדת תחילת המשפט, IntelliSense מציגה רשימת ערכים חוקיים התואמים לטקסט שהוקלד

להלן שיגרה פשוטה הכוללת שלוש שורות. השורה הראשונה מקצה מיקום בזיכרון לתוצאה מחושבת. השנייה מחברת שני קבועים. השורה השלישית מדפיסה את התוצאה בחלון **Immediate** (מיידי). חלון זה משמש כפנקס לאחסון תוצאות ביניים במהלך בדיקת הקוד. כדי להציגו בחר **View** (תצוגה), **Immediate Window**. ניתן להפעיל את השיגרה מתוך עורך VB על ידי לחיצה במקום כלשהו בשיגרה ואחר כך לחיצה על הלחצן **Run Sub/UserForm** שבסרגל הכלים הרגיל.

```
Sub MyFirstCalculator()
Dim Result
   Result = 1 + 2
   Debug.Print Result
End Sub
```

במצבים אופייניים יותר, תוכל לקרוא לשיגרה באחת משתי דרכים. תוכל לכלול את שמה בשורה עצמאית; אם השיגרה כוללת ארגומנטים כלשהם, תוכל לכלול אותם לאחר שמה, כשהם מופרדים בפסיקים. לחילופין, תוכל להקדים את מילת המפתח **Call** לשם השיגרה. Call היא מילת מפתח של VBA המשמשת לקריאה לשיגרה. אם תשתמש במילה Call, תחום בסוגריים את הארגומנטים המופיעים לאחר שם הפונקציה.

להלן גירסה גמישה מעט יותר של פונקציית המחשבון הבסיסית. בדוגמה מופיעות שתי שגרות. השיגרה שנקראת MySecondCalculator מחברת שני מספרים, ומדפיסה את התוצאה בחלון Immediate. השיגרה קולטת את המספרים באמצעות שני ארגומנטים שנשלחים אליה. השיגרה השנייה קוראת לשיגרה שמבצעת את פעולת החיבור. תוכל לשנות את המספרים שהשיגרה מסכמת, על ידי שינוי ערכי הארגומנטים שבשיגרה הראשונה. בגירסה מתוחכמת מעט יותר של יישום זה, אפשר לקשור את ערכי הארגומנטים למשתנים או לשדות טופס.

```
Sub CallSecondCalculator()
   MySecondCalculator 1, 3
End Sub

Sub MySecondCalculator(First, Second)
Dim Result
   Result = First + Second
   Debug.Print Result
End Sub
```

שגרות פונקציה

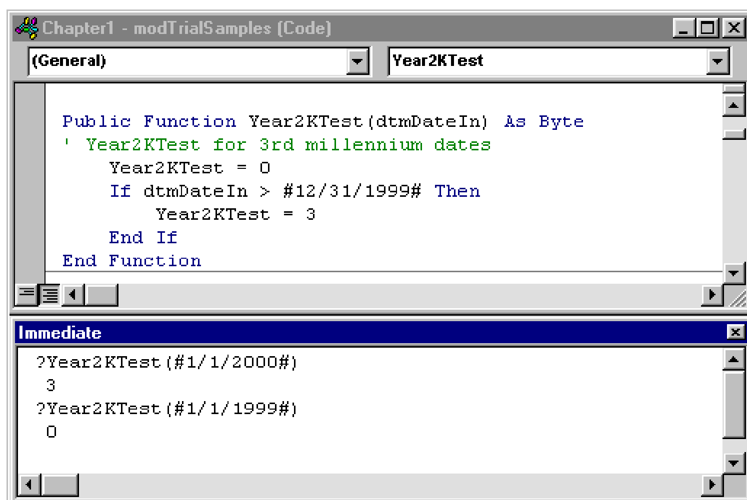
שגרות פונקציה, הנקראות בדרך-כלל **פונקציות** (Functions), שונות משגרות רגילות במספר מובנים. ראשית הן מחזירות ערך, ולכן ניתן לנצלן בביטויים כאילו היו משתנים. שנית, הן אינן משמשות כשגרות אירוע. שגרות ופונקציות יכולות לבצע מטלות. למעט הנקודות שצוינו, ניתן לנצל את שני הסוגים לביצוע מטלות זהות רבות.

פונקציה היא אוסף משפטי VBA, תחומים בין משפטי **Function** ו- **End Function**. פונקציה יכולה לקבל ארגומנטים בדיוק כמו שיגרה. היא יכולה להכיל ביטוי אחד או יותר, שלפחות אחד מהם יכול לקבוע את הערך שיוחזר בשם הפונקציה. אפשר לסיים פונקציה לאחר קביעת ערכה, באמצעות המשפט **Exit Function**. כל פונקציה יכולה להכיל כמה משפטי Exit Function.

פונקציה מסוגלת להחזיר ערך, אך אינה חייבת לעשות כך. פונקציה יכולה להיות אוסף משפטים המפעילים שיטות ומגדירים מאפיינים, מבלי להחזיר ערך. זו אחת התכונות המשותפות לפונקציות ולשגרות.

אתה מתחיל פונקציה ושיגרה באותה דרך, אך בתיבת הדו-שיח **Add Procedure**, יש לבחור באפשרות **Function** במקום באפשרות **Sub**. ניתן להפעיל פונקציה על ידי לחיצה על הלחצן Run Sub/UserForm שבסרגל הכלים Standard של VBE. תוכל להפעיל פונקציה גם מתוך החלון **Immediate**: הקלד סימן שאלה ואחריו שם פונקציה. אם הפונקציה כוללת ארגומנטים, יש להציבם בסוגריים ולהפרידם בפסיקים. מתוך חלון זה תוכל להפעיל פונקציות שכתבת בעצמך וגם פונקציות מוכללות של Access.

תרשים 1.5 מציג את VBE עם פונקציה פשוטה הקובעת אם התאריך המופיע כארגומנט, חל במילניום השלישי. כדי לעמוד בכללים המקובלים, נתייחס למילניום זה כאילו הוא מתחיל בשנת 2000 במקום בשנת 2001. הפונקציה Year2KTest קולטת תאריך ומחזירה 3 אם התאריך חל במילניום השלישי, או 0 בכל מקרה אחר. החלון Immediate המוצג מתחת לחלון Code, מציג את תוצאת הפעלת הפונקציה על שני תאריכים שונים. החלון Immediate שבתרשים 1.5 מאשר את התוצאה על ידי החזרת הערך 3 לארגומנט שהוא היום הראשון בשנת 2000, ואת הערך 0 עבור היום הראשון בשנת 1999. שים לב כי עליך לתחום את התאריך בין תווי #.



תרשים 1.5: פונקציה פשוטה שמופעלת מתוך החלון Immediate

הדוגמה הבאה מציגה גישה מתוחכמת יותר לזיהוי המילניום. הפונקציה פותחת בהצהרה על טווח התאריכים המדויק. לאחר מכן היא בודקת את הארגומנט כנגד שני תאריכי ציון. אם התאריך אינו חל במילניום הראשון או השני, הפונקציה מניחה שהוא חל במילניום השלישי. משפט תיבת ההודעה ההתחלתי מציין את טווח התאריכים שבו הפונקציה מחזירה תוצאות מדויקות.

```
Public Function PopularMillenium(dtmDateIn) As Byte
    MsgBox "This works for dates after 12/31/0099 and before 1/1/3000.", _
        vbInformation,  "Programming Microsoft Access 2000"

    If dtmDateIn <= #12/31/999# Then
        PopularMillenium = 1

    ElseIf dtmDateIn <= #12/31/1999# Then
        PopularMillenium = 2

    Else
        PopularMillenium = 3

    End If

End Function
```

טווח התאריכים החוקי של Access 2000 הוא 1/1/100 עד 31/12/2999. טווח זה מספק את דרישות מרבית היישומים השולחניים. אם תידרש לטפל בתאריכים החורגים מטווח זה, שקול תכנות תאריכים בנפרד ממערכת התאריך הסדרתי של Access.

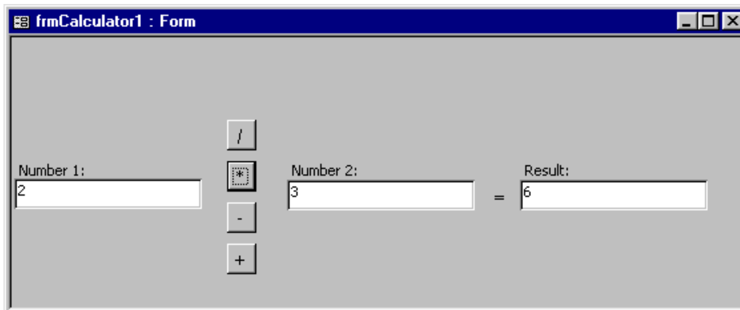


הערה:

Access 2000 תואם לשנת 2000. כמו גרסאותיו הקודמות, הוא מאחסן את ערך השנה בארבע ספרות. היישום גם מטפל בצורה תקינה בשנים מעוברות: שנה שמתחלקת ב-4 היא שנה מעוברת, אלא אם היא מתחלקת ב-100. לעומת זאת, שנה שמתחלקת ב-400 היא כן שנה מעוברת. שנת 2000 מתחלקת ב-400, ולכן היא שנה מעוברת. כלל זה הוא קריטי כשמחשבים את ההפרש בין שני תאריכים. תבנית התאריך הכללי ותבנית התאריך הקצר של Access מנצלות את אפשרויות תבנית התאריך הקצר של מערכת ההפעלה, כדי לקבוע כיצד להציג את התאריך בצורה תקינה. אם תשנה את ההגדרות האזוריות שבלוח הבקרה להצגת השנה באמצעות ארבע ספרות, כל תבניות התאריך הכללי תצגנה את השנה כך.

החוקים שתוארו אינם כופים עליך לפתח יישומים באופן שמבטיח אותם מפני בעיית שנות 2000. למרות ש- Access 2000 תואם לשנת 2000, כל יישום שלו מועד לבעיה. בקר באתר Microsoft: (www.microsoft.com/technet/topics/year2k/default.htm) לקבלת סקירה על הנושא, ומאמרים ייחודיים על מוצרים והתאמתם לשנת 2000. אתר FMS (www.fmsinc.com/tpapers/index.html#Year 2000 Papers) מציג עמדה שונה בסוגיה זו. FMS משווקת מוצר שנקרא Total Access Inspector שאמור לזהות בעיות תאימות שנת 2000 ביישומים מבוססי Access.

תוכל להשתמש בשגרות ובפונקציות כדי לפתח פתרון לבעיה. תרשים 1.6 מציג טופס שמבוסס על שגרות ופונקציות. הטופס מאפשר למשתמש להקליד ערכים בתיבות הטקסט שתוויותיהן Number 1 ו- Number 2. לחיצה על לחצני /, *, – או +, גורמת לחישוב התוצאה והצגתה בתיבת הטקסט Result.



תרשים 1.6: טופס זה משמש כמחשבון פשוט. פונקציות VBA מאפשרות שימוש בלחצני הטופס ומציגות את התוצאה בתיבת הטקסט Result בהתאם לערכים שבשתי תיבות הטקסט האחרות

קוד VBA של הטופס שבתרשים 1.6 מנצל רק ארבעה זוגות שגרות, כפי שמתואר בהמשך. ארבע שגרות מטפלות באירועי הלחיצה על לחצני האופרטורים לחישוב. הטיפול באירועים ממוצה בקריאה לפונקציה ששולפת ערכים מתוך שתי תיבות

טקסט, מבצעת פעולת חישוב באמצעות האופרטור שמצוין על הלחצן שנלחץ, ומחזירה את הערך שחושב לשגרת האירוע. בתגובה, מציבה שגרת האירוע את הערך המוחזר מהפונקציה בתיבת הטקסט השלישית בטופס. שים לב לשימוש האופציונלי בקידומת Me לפני שמות תיבות הטקסט. משום שהקוד הינו קוד הטופס שכולל את תיבת הטקסט, אין חובה לציין את שם הטופס. שגרות האירוע מנצלות את הקידומת Me במקום שם המחלקה הארוך והרשמי של הטופס, Form_frmCalculator1.

```
Option Compare Database
Option Explicit
Dim dblResult As Double
Private Sub cmdAddition_Click()
   Me.txtResult = MyAdder
End Sub
Private Function MyAdder()
   dblResult = CDbl(txtNumber1) + CDbl(txtNumber2)
   MyAdder = dblResult
End Function

Private Sub cmdSubtraction_Click()
   Me.txtResult = MySubtractor
End Sub
Private Function MySubtractor()
   dblResult = CDbl(txtNumber1) - CDbl(txtNumber2)
   MySubtractor = dblResult
End Function

Private Sub cmdMultiplication_Click()
   Me.txtResult = MyMultiplier
End Sub
Private Function MyMultiplier()
   dblResult = CDbl(txtNumber1) * CDbl(txtNumber2)
   MyMultiplier = dblResult
End Function

Private Sub cmdDivision_Click()
   Me.txtResult = MyDivider
End Sub
Private Function MyDivider()
   dblResult = CDbl(txtNumber1) / CDbl(txtNumber2)
   MyDivider = dblResult
End Function
```

הפונקציות והשגרות נמצאות במודול הקוד של הטופס. המשפט Dim שבראש המודול מצהיר על משתנה שכל שגרות המודול יכולות לגשת אליו. המשתמשים יכולים ללחוץ על לחצן אופרטור אחד בלבד בזמן נתון, ולכן ניתן לשתף את dblResult. משפט

Option Explicit כופה הצהרה על משתנים בטרם שימוש בהם, כדי להימנע משגיאות דפוס המהוות מקור שכיח לשגיאות בתוכנית. **Option Compare Database** הוא מפרט ברמת-מודול המקצה משתני מחרוזת ממוינים בסדר שנקבע בלוח הבקרה.

זוג השגרות הבא מדגים את רוב הנושאים הקשורים לשגרות, פונקציות ושיטות בהם עסקנו בתחילת הפרק. הטופס frmCalculator2 כולל שני פקדים בלבד: תיבת טקסט שנקראת txtInput ולחצן פקודה שנקרא cmdSquarer. כשהמשתמש לוחץ על הלחצן, הטופס מחשב את השורש הריבועי של הערך שבתיבת הטקסט. השיגרה מציגה את התוצאה בתיבת הודעה.

```
Option Compare Database
Option Explicit
Dim dblResult As Double
Private Sub cmdSquarer_Click()
   MySquarer Form_frmCalculator2.txtInput
End Sub
Public Sub MySquarer(MyOtherNumber As Double)
   dblResult = MyOtherNumber * MyOtherNumber
   MsgBox dblResult
' Optional statements illustrating the use of methods
'    DoCmd.GoToControl "txtInput"
'    txtInput.SetFocus
'    DoCmd.Close acForm, "frmCalculator2", acSaveNo
End Sub
```

שגרת האירוע cmdSquarer_Click מפעילה את השיגרה MySquarer ומעבירה לה כארגומנט את תוכן תיבת הטקסט txtInput. MySquarer מחשבת את התוצאה ומציגה אותה בתיבת הודעה.

שלוש שורות ההערה האחרונות מציעות פעולות נוספות לביצוע. השורה הקוראת לשיטה GoToControl מתארת כיצד להעביר את המיקוד מהלחצן אל תיבת הטקסט. השורה הבאה (SetFocus) מציעה דרך חלופית להשיג תוצאה זהה. השיטה Close מתארת סגירת טופס. שים לב ששורה זו מציינת שם נוסף frmCalculator2, באמצעותו היא מפנה לטופס. יש חשיבות לשימוש בקבוע acSaveNo מכיון שהוא מאפשר לסגור טופס ללא בקשת אישור שמירה מהמשתמש.

שגרות מאפיין

שגרות מאפיין משמשות להגדרת מאפיינים מותאמים אישית לטפסים, דוחות ומודולי מחלקה. בסעיף הבא נעסוק במודולי מחלקה, בפרק 7 נציג דוגמאות לשגרות מאפיין.

יש שלושה סוגי משפטי מאפיין: **Property Get**, **Property Let** ו- **Property Set**. תוכל לנצל משפטים אלה להוספת מאפיינים מיוחדים לטופס. המשפט Property Get ובן זוגו End Property יכולים להחזיר ערך, בדיוק כפי שעושה פונקציה. אם מגדירים

מאפיין בעזרת משפט Property Get בלבד, המאפיין יהיה לקריאה בלבד. מאפיין כזה נוח לשימוש אם אתה יכול להציג ערך כלשהו, אך לא לשנותו. למשל ציוני מבחנים.

קיימים מאפיינים שלא ניתן לקרוא, אך יש חשיבות ליכולת לשנותם. לדוגמה, מנהלי אבטחת מסדי נתונים אינם זקוקים ליכולת לקרוא סיסמאות משתמשים; הם רק צריכים להיות מסוגלים לדרוס אותן כשהמשתמשים שוכחים את סיסמת הכניסה שלהם. המשפטים Property Let ו- End Property יכולים לשמש להגדרת סיסמה.

המשפט Property Set דומה למשפט Property Let בכך ששניהם משמשים להגדרת מאפיין. המשפט Property Let מגדיר מאפיין ששווה לסוג נתונים, כגון מחרוזת (string) או מספר שלם (integer). המשפט Property Set מגדיר מאפיין ששווה להפניה לאובייקט. משתמשים במשפט Property Set בצירוף הפניות לאובייקטים, כגון הפניה לטופס או לדוח.

מאפיינים רבים הם לקריאה ולכתיבה גם יחד, ולכן לעיתים קרובות משתמשים במשפט Property Get יחד עם המשפט Property Let או Property Set. במקרה זה, על זוג משפטי Property להיות בעלי שם זהה, כדי שיוכלו להפנות לאותו מאפיין.

מודולים

מודול הוא מכולה של שגרות והצהרות, כגון Option Explicit ו-Dim. קיימים שני סוגי מודול בסיסיים. הסוג הראשון הוא מודולים סטנדרטיים המוצגים ברשימה לאחר לחיצה על הלחצן **מודולים** (Modules) שבחלון **מסד נתונים** (Database). השגרות ב**מודול סטנדרטי** אינן תלויות באובייקטים הקיימים בקובץ מסד נתונים של Access. משמעות הדבר, שאין הפניות אל Me או שמות פקדים ללא קידומות מתאימות של הפקדים. לעומת זאת, יישומים יכולים לפנות לשגרות במודולים סטנדרטיים מתוך כל אובייקט אחר.

הסוג השני הוא **מודול מחלקה** (Class Module) ובו שלושה טיפוסים בסיסיים: **מודולי מחלקת טופס**, **מודולי מחלקת דוח** ו**מודולי מחלקה מותאמים אישית**. עקרונית, מודולים יכולים לגשת לשגרות שנמצאות במודולים אחרים. ניתן להוציא שיגרה מחוץ ל**טווח ההכרה** (scope) הכללי על ידי השימוש במילת המפתח **Private** (פרטי) בעת הגדרת השיגרה (תרשים 1.7 מציג דוגמה לתחביר). תוכל גם להצהיר מפורשות על שגרות כבעלות טווח הכרה גלובלי על ידי שימוש במילת המפתח **Public** (ציבורי).

מודול טופס הוא מודול של טופס כלשהו שכולל הצהרה או שיגרה אחת לפחות. יצירת שגרת אירוע של טופס או פקד בטופס, גורמת ליצירת מודול מחלקת טופס. מודולי מחלקת דוח פועלים בצורה דומה למודולי מחלקת טופס, אך אירועי **דוח** שונים מאירועי טופס, ובדרך-כלל לא יופיע צירוף פקדים זהה בטופס ובדוח. תוכל ליצור מודולי מחלקה מותאמים אישית המכילים פונקציות שיטה ופונקציות שיגרה של אובייקט כלשהו, כגון עובד או חשבון בנק. תוכל לייחס את השיטות והמאפיינים למודולי מחלקה מותאמים אישית, כפי שניתן לעשות עם מחלקות מוכללות של Access.

השימוש במודולי מחלקה מותאמים אישית דומה לשימוש בשבלונה ומטרתו ליצור מופעים (instances) חדשים של מחלקה. Access מאפשר לעשות זאת בשתי דרכים. הראשונה, באמצעות משפט Dim אחד שמגדיר את המחלקה וגם יוצר מופע חדש של המחלקה שהוגדרה. להלן תחביר המשפט:

```
Dim objInstance As New objClass
```

הגישה השנייה מבוססת על שני משפטים. הראשון מצהיר על מופע האובייקט. השני מגדיר הפניה אל האובייקט. להלן תחביר המשפטים:

```
Dim objInstance as objClass
Set objInstance = New objClass
```

השם objClass פונה למודול מחלקה הכולל שגרות מאפיין ופונקציות שיטה ציבוריות. פונקציות שיטה אלו פועלות כשיטות מחלקה, בדיוק כפי ששגרות מאפיין משמשות להגדרת מאפיינים. פרק 7 מתאר כיצד יוצרים ומשתמשים במודולי מחלקה מותאמים אישית.

ממשק VBE

השינוי הבולט ביותר בסביבת הפיתוח של Access הוא הממשק החדש של VBE. מנקודת המבט של ממשק הפיתוח, ממשק VBE מגביר את התאימות בין Access 2000 ליישומי Office האחרים (Word, Excel ו-PowerPoint). בסעיף זה נבחן לעומק את תכלית החלונות ופריסתם וכיצד משתמשים בהם לניפוי שגיאות ופעולות נוספות. כמו-כן נסקור את השימוש בסורק האובייקטים (Object Browser).

חלונות VBE

Access 2000 מציע לפחות שלושה נתיבים אל VBE של מודולים שאינם תומכים בטופס או בדוח. מי שיש לו ניסיון בפיתוח באמצעות רכיבי Office אחרים, יכול לבחור בחלון **מסד נתונים** (Database) את תפריט **כלים** (Tools), **מאקרו** (Macro), **Visual Basic Editor**. קיצור הדרך מהמקלדת הוא **Alt+F11**. מקשי הקיצור מאפשרים גם להחליף בין עורך VB לחלון מסד הנתונים. אפשרות נוספת, אם ברשותך מודולים סטנדרטיים כלשהם, לחץ על לחצן **מודולים** (Modules) שבחלון **מסד נתונים**, ואחר כך לחץ לחיצה כפולה על מודול סטנדרטי שברצונך להציג. חלון VBE ייפתח כשהוא מציג את המודול שבחרת. כדי ליצור מודול סטנדרטי חדש, יש ללחוץ על לחצן **מודולים** בחלון **מסד נתונים** ולאחר מכן על **חדש** (New) בסרגל הכלים. פעולה זו תפתח מודול חדש וריק.



הערה:

צוות הפיתוח של Office עובד על פיתוח נתיב הפעלה רביעי של VBE שיהיה מוכר לחלק מהמפתחים לסביבת Access. הנתיב החדש יאפשר לבחור, טופס או דוח, או מודול סטנדרטי בחלון **מסד נתונים** וללחוץ על לחצן **קוד** (Code). לחצן זה יפתח את VBE כשהוא מציג את הקוד התואם לאובייקט הנבחר.

כדי להגיע אל מודול התומך בטופס או דוח, עליך לפתוח תחילה את האובייקט בתצוגת **עיצוב** (Design). תוכל ללחוץ על לחצן **קוד** (Code) שבסרגל הכלים **עיצוב** (Design). חלון VBE ייפתח ויציג את ראש המודול. כדי לעבור לשגרת האירוע המטפלת באובייקט מוגדר בטופס או בדוח, לחץ על הלחצן **בניה** (Build) בשורת המאפיין הרצוי, בכרטיסיה **אירוע** שבגיליון המאפיינים. אם לא קיימת שגרת אירוע לאירוע הרצוי של האובייקט, הלחיצה על לחצן **בניה** ובחירה ב**בונה קוד** (Code Builder) תפתח שגרת אירוע ריקה.

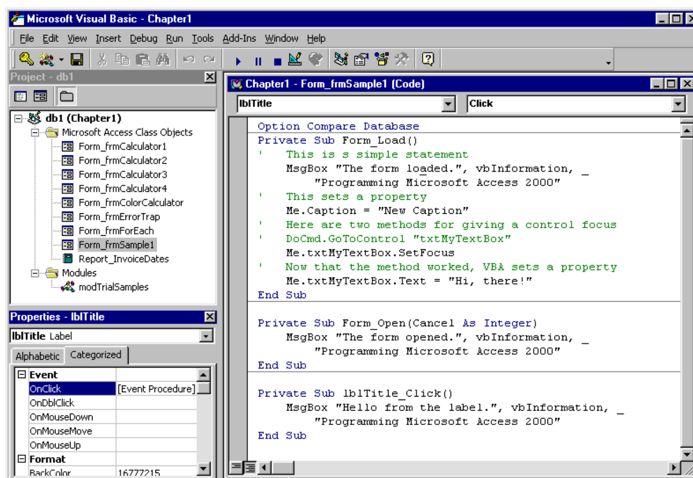
כשתגיע ל-VBE, תרצה בוודאי לפתוח את החלונות **Project** (המכונה גם Project Explorer) ו- **Properties**. חלונות אלה מאפשרים לפתוח ולבחון מודולים נוספים של היישום בצורה נוחה. החלון **Project** מציג מודולים שאינם תומכים בטופס או בדוח ונמצאים בתיקיה **Modules**. מודולים שתומכים בטפסים ובדוחות נמצאים בתיקיה **Microsoft Access Class Objects**. כדי להציג ולהגדיר את האובייקטים הקשורים למחלקה, תוכל לבחור תיקיה הקשורה לטופס או לדוח. לפתיחת חלונות אלה, בחר בפקודה המתאימה מתוך תפריט **View** (תצוגה), או השתמש במקלדת או בלחצני סרגל הכלים.



הערה:

כדי שהחלון **Properties** יוכל להציג את האובייקטים של מחלקת אובייקט ב-Access, אובייקט זה חייב להיות פתוח בתצוגת **עיצוב** (Design). לחיצה כפולה על מודול או על מחלקה בחלון **Project** מציגה את ההוראות וההצהרות המתאימות בחלון **Code** התואם, אך האובייקטים מופיעים רק בתנאי שאובייקט המחלקה גם פתוח בתצוגת **עיצוב**.

תרשים 1.7 מציג את קוד מסד הנתונים המדגמי של פרק זה, לאחר טעינתו ל-VBE. חלון **Project** מציג את המחלקה Form_frmSample1 שנבחרה. תחת חלון זה ניתן להבחין בחלון **Properties** המציג את פקד התווית של כותרת הדוח שנבחר. החלון מראה גם שלפקד התווית שיגרה הקשורה לאירוע OnClick. החלון **Code** שמימין לחלונות **Project** ו-**Properties** מציג את קוד שגרת האירוע. באפשרותך להזיז, לשנות גודל או לעגן כל אחד מחלונות אלה.

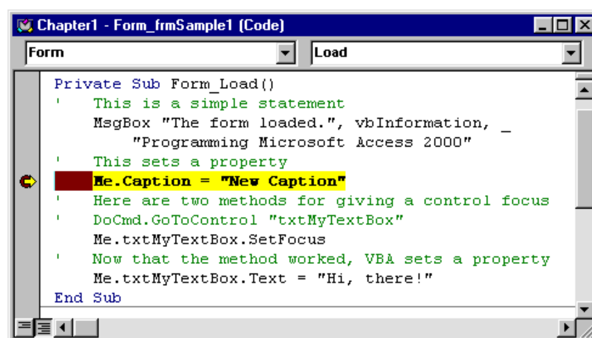


תרשים 1.7: החלונות **Code**, **Project** ו-**Properties** מציגים את קוד מסד הנתונים המדגמי של פרק זה

לחיצה כפולה על מחלקה אחרת או אובייקט Module כלשהו של Access בחלון **Project** פותחת את חלון הקוד התואם. החלון מאפשר לבדוק, לערוך או להעתיק את הקוד. חלון **Code** מעוצב בצורה מוכרת, בחלקו העליון תיבות הרשימה הנפתחת **Object** ו-**Procedure**. ניתן לנצל אותו כדי לנווט במודול גדול, או כדי לפתוח שגרות חדשות במודול קיים. החלון **Properties** מציג את מאפייני המחלקה או המודול הנוכחי של Access. החלון מאפשר לערוך את מאפייני האובייקט, אולם קל ונוח יותר לערוך מאפייני אובייקט של טופס או דוח בתצוגת **עיצוב**.

ניפוי שגיאות

תוכל לנצל את חלון **Code** לבדיקת הקוד ולניפוי שגיאות. תוכל להוסיף או לבטל נקודת עצירה (breakpoint) במשפט, על ידי לחיצה על השוליים שלשמאלו. VBE מסמן את נקודות העצירה בדרך המקובלת, כלומר נקודה בשוליים השמאליים.



תרשים 1.8: חלון **Code** של אירוע טעינת המחלקה Form_frmSample1 בעת עצירה. החץ מצביע על שורת הקוד הבאה לביצוע

תרשים 1.8 מציג את מאתר הבאגים בעצירה על השורה השנייה מתוך 4 שורות הקוד בשיגרה; החץ שבשוליים השמאליים מצביע על שורת הקוד הבאה לביצוע.

לחידוש ביצוע הקוד לאחר העצירה, לחץ על **Continue** (המשך) בתפריט **Run** (הפעלה). תרשים 1.9 מציג את תוצאת ביצוע שאר חלקי הקוד של השיגרה Form_frmSample1. שים לב שהכיתוב בטופס הוא "New Caption". הטקסט בתיבת הטקסט הוא "Hi, there!".



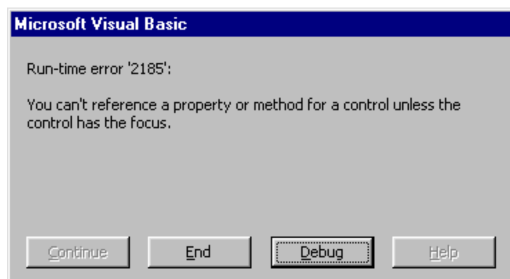
תרשים 1.9: תוצאת ההפעלה של שלוש שורות הקוד הנותרות של תרשים 1.8

תוכל להשתמש בחץ שבתרשים 1.8 כדי לדלג על שורת קוד אחת או יותר בנתיב הביצוע. לדוגמה, תרשים 1.10 מציג את תוצאות גרירת החץ לשורה שמפעילה את השיטה SetFocus ובחירה בפקודה **Continue** מתוך התפריט **Run**. השיגרה מדלגת על הצבת המחרוזת "New Caption" במאפיין Caption (כיתוב) של הטופס, ולכן הטופס נראה כמעט זהה לזה שבתרשים 1.9, למעט הכיתוב שלו, "My Default Caption".



תרשים 1.10: תוצאת הדילוג על השורה השנייה הנותנת בתרשים 1.8 והמשך הביצוע

תרשים 1.11 מציג את תוצאת גרירת החץ בתרשים 1.8 אל שורת הקוד האחרונה שמבצעת את ההצבה בתיבת הטקסט. פעולה זו תגרום כמובן, להצגת הודעת שגיאה, עקב הניסיון להציב ערך במאפיין מבלי להעביר את המיקוד לאובייקט שלו.

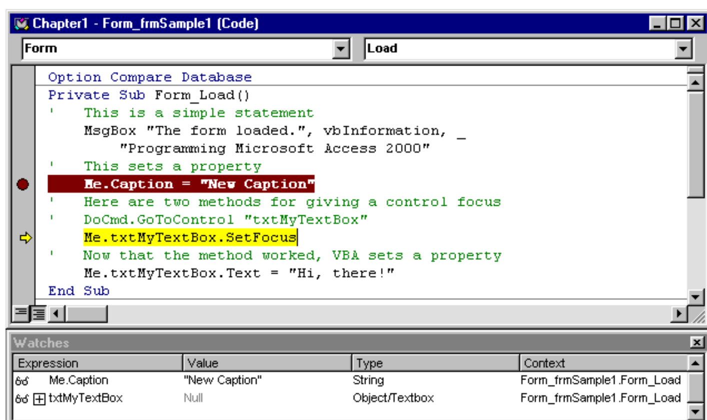


תרשים 1.11: תוצאת הדילוג אל השורה האחרונה של תרשים 1.8 והמשך ביצוע הקוד

Access 2000 כולל את רוב הרכיבים התפקודיים של חלון איתור הבאגים של Access 97, שהכיל חלון Watch וחלון Locals. כל חלון הכיל מסך מפוצל שכלל את החלון המיידי (Immediate). Access 2000 מציג את שלושת החלונות, **Watch**, **Locals** ו-**Immediate** בצורה נפרדת. ב-VBE פותחים את החלונות על ידי שימוש בתפריט **View** (תצוגה). תוכל לגרור, לשחרר ולשנות גודל של כל אחד מהחלונות לצד החלונות **Code**, **Project** ו-**Properties**. בחר את **Tools** (כלים), **Options** (אפשרויות) והשתמש בכרטיסיה **Docking** (עיגון) כדי לקבוע את החלונות שברצונך לעגן.

תוכל לנצל את חלון **Watch** (צפייה) למעקב אחר ערכי ביטויים, משתנים ואובייקטים במהלך ביצוע הקוד. לאחר פתיחת חלון Watch, תוכל להוסיף משתנים לצפייה על ידי בחירה ב-**Debug**, **Add Watch**. בטרם תפעיל את הפקודה, בחר משתנה למעקב. כשתיפתח תיבת הדו-שיח **Add Watch**, בחר סוג לצפייה ולחץ על **OK** כדי לסגור את תיבת הדו-שיח. ערך המשתנה מוצג בחלון הצפייה במהלך ביצוע הקוד. אם תבצע את הקוד במצב עצירה (break mode), תוכל לאמת ערך של משתנים חיוניים בכל שלב ביצוע.

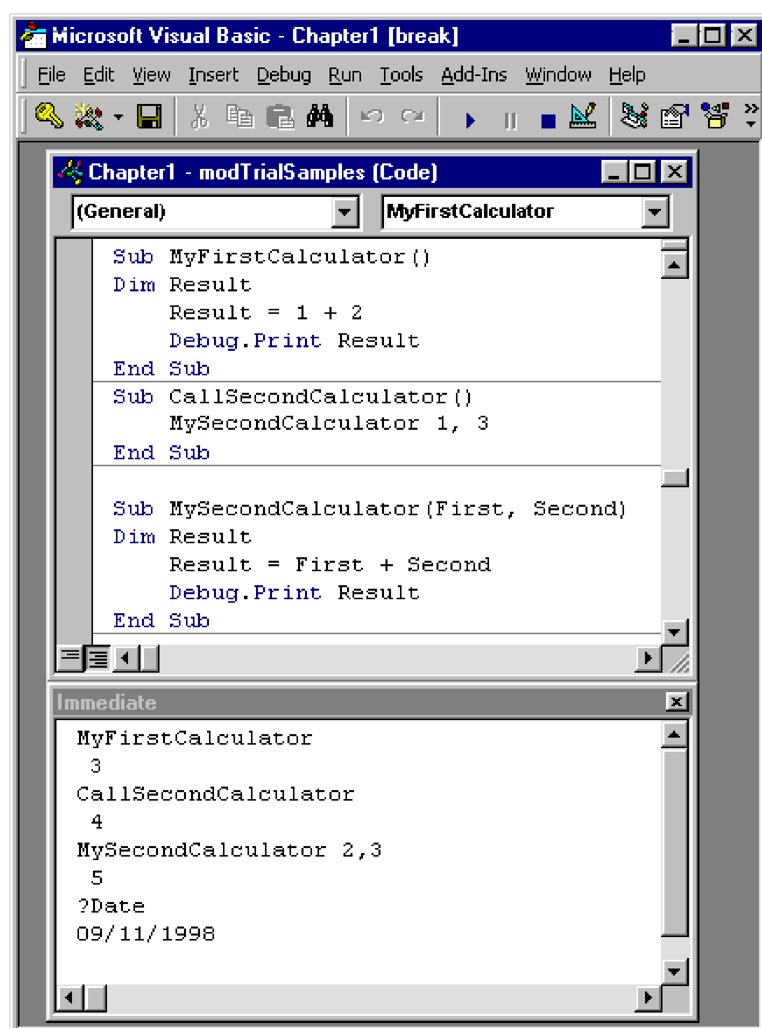
תרשים 1.12 מציג את ערך המאפיין **Caption** של frmSample1, מייד לאחר שהקוד שינה אותו מ"My Default Value" ל"New Caption". חלון Watch מציג את ערך txtMyTextBox בתור Null, מכיון שהקוד לא ביצע עדיין את משפט ההצבה.



תרשים 1.12: חלון Watch המציג את מצב הביטויים בעת ביצוע שגרת אירוע

החלון Locals יכול להציג את כל המשתנים שבטווח ההכרה, בעת שהקוד מתבצע במצב עצירה. כשמגיעים לנקודת העצירה, החלון מכיל אובייקט Me. תוכל להרחיב את Me ואת רכיביו שנבחרו, בזה אחר זה, כדי לגלות את ערכי כל המאפיינים והמשתנים. הדבר יסייע לך כשיש צורך לחקור את מהלך התוכנית לעומק.

החלון **Immediate** עשוי להיות הכלי הנוח ביותר לפיתוח קוד וניפוי באגים. החלון מאפשר להפעיל פונקציה או שגרת משנה כלשהי שנמצאת בטווח ההכרה. תוכל גם להעריך ביטויים ולבדוק דרכים שונות לכתיבת פונקציות. בגרסאות שלא איפשרו להציג ביטויי צפייה, החלון המיידי היה מקום נוח להדפסת תוצאות ביניים במהלך ביצוע תוכנית במצב עצירה. ניתן עדיין לנצל חלון זה בנסיבות מיוחדות.



תרשים 1.13: כך נראה חלון **Immediate** בעת הפעלת שגרות ופונקציות מוכללות

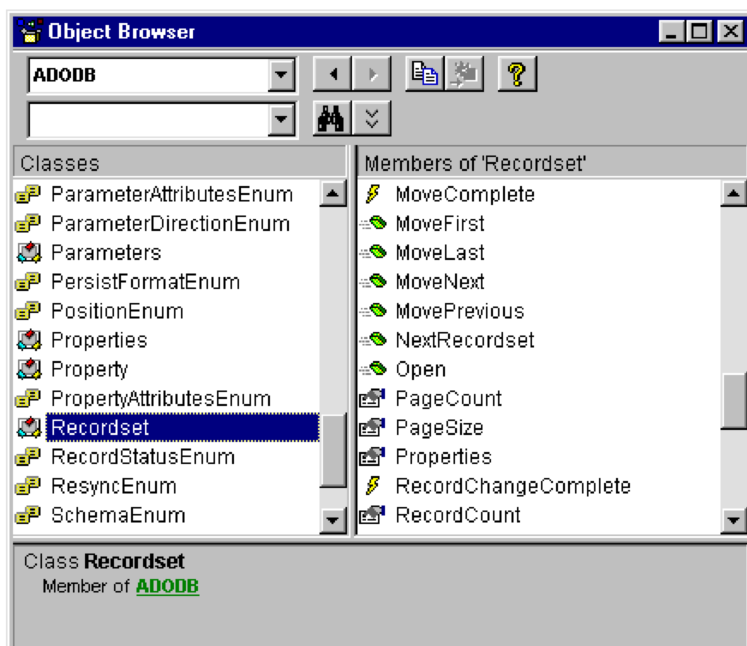
תרשים 1.13 מציג שלושה שימושים אפשריים של החלון **Immediate**. עסקנו בפונקציות אלו בתחילת הדיון בשגרות, אך לא הצגנו את הפלט שלהן בחלון **Immediate**. הקלדת MyFirstCalculator והקשת Enter בחלון **Immediate**, מפעילה שיגרה שנקראת בשם זה. מופעלת הפונקציה שמדפיסה את התוצאה 3 בחלון. כך גם בשיגרה CallSecondCalculator, והתוצאה במקרה זה היא 4. הדוגמה האחרונה מציגה כיצד מציינים ארגומנטים עבור הפונקציה MySecondCalculator. כל שעליך לעשות הוא להקליד את שמה, רווח ואת הארגומנטים כשהם מופרדים בפסיקים. לדוגמה זו חשיבות מכרעת, כיון שהיא ממחישה את אופן השימוש בארגומנטים המועברים לשיגרה או פונקציה כדי לחשב ערך החזרה.

תוכל גם לנצל את חלון **Immediate** להפעלת פונקציות מוכללות ומותאמות אישית. בעבודה עם פונקציה, עליך לציין פקודת הדפסה כדי להחזיר את תוצאת הפונקציה. באפשרותך להקליד את מילת המפתח Print, ללחוץ על לחצן **?**, או להקליד את שם הפונקציה ולסיים בהקשה על Enter. הדוגמה האחרונה שבחלון **Immediate** מציגה כיצד קוראים לפונקציה המוכללת Date.

סורק האובייקטים

Object Browser (סורק האובייקטים) המוצג בתרשים 1.14, הוא כלי רב-עוצמה ללימוד נושא מודלי אובייקט. הכלי חשוב במיוחד למפתחים בסביבת Access, עקב החידושים הרבים שהוא כולל בנושא מודל האובייקט. לדוגמה, Access 2000 כולל שפת פיתוח חדשה לגישה לנתונים – ADO (ActiveX Data Objects) – שתדחק בסופו של דבר את מקומה של DAO (Data Access Objects) ממקומה. מימוש ADO ב- Access 2000 מבוסס על לפחות שלושה מודלי אובייקט.

בטרם תציג את מחלקות האובייקט של ADO בסורק האובייקטים, עליך להגדיר הפניה אל הספריה המתאימה. בחר **Tools** (כלים), **References** (הפניות) כדי לאמת או ליצור קישורים אל ספריות מסוג ADO. פתח את סורק האובייקטים על ידי לחיצה על הלחצן Object Browser בסרגל הכלים הרגיל של VBE. ניתן לפתוח אותו גם באמצעות תפריט **View** או הקשה על מקש הקיצור F2. אחת הספריות ADO נקראת ADODB. בחר אותה מתוך הרשימה הנפתחת Project/Library בראש סורק האובייקטים. בחירת הספריה משנה את תוכן הרשימות **Classes** (מחלקות) ו-**Members** (חברים) של הסורק. תרשים 1.14 מציג את המחלקה Recordset שנבחרה מתוך הרשימה **Classes** ורשימה מעורבת של מאפיינים, שיטות ואירועים המוצגים ברשימה **Members**. בחר ערך של חבר ולחץ על לחצן **?** כדי לקבל עזרה מפורטת בנושא. הודות למנגון החיפוש שלו, סורק האובייקטים מסוגל גם לחפש מחלקות וחברים מוגדרים. השתמש בתיבת הרשימה הנפתחת השנייה כדי לציין קריטריוני חיפוש. לעיתים מקבלים כך מחלקות וחברים נוספים שנמצאים בספריות שונות. סורק האובייקטים מאפשר לבדוק אותם בנפרד.



תרשים 1.14: מחלקת ערכת הרשומות של ADODB ומבחר מחבריה, כפי שהם נראים בתצוגת Object Browser

Jet, סוגי נתונים והצהרות

ב-Access עליך לציין סוגי נתונים במקומות רבים, לרבות משתנים בשגרות וטבלאות נתונים שמאוחסנות בהתקני אחסון. הסעיפים הבאים עוסקים בנושאי פיתוח הקשורים לנתונים שמקורם ב- Jet 4 ובשימוש במשתנים בשגרות.

Jet

Access 2000 תומך ביסודו בשני מנגנוני מסד נתונים. במקור, Access היה קשור למנגנון מסד הנתונים Jet. Access 2000 כולל גם תמיכה מוכללת במסד הנתונים SQL Server 7. פרק 12 עוסק בהיבטי התאימות של Access 2000 עם SQL Server. הסעיף הנוכחי עוסק בהשפעת חידושי Jet 4 על ביצוע יישומי Access 2000. חידושים אלה מסוגלים להשפיע על גודל קבצי מסד הנתונים ועל נעילת רשומות, על אופן הטיפול בסוגי נתוני שדות שנבחרים (כולל שדות המפתח Memo (תזכיר), Hyperlink (היפר-קישור) ו- auto-incrementing (מספרי אוטומטי בהפרשים קבועים) וקישורים אל מקורות נתונים חיצוניים.

Access 2000 מאחסן את כל נתוני הטקסט וההערות בתבנית Unicode. תבנית זו מחליפה את מערך התווים מרובה הבתים – MBCS (multi-byte character set) שהיה מקובל בגרסאות קודמות של Access, לטיפול בשפות כגון יפנית, סינית ועוד. Unicode הוא תקן-משנה של ISO 10646, המתאר את שיטת הקידוד הדיגיטלית של כל השפות הכתובות. תקן חדש זה מייצג כל תו באמצעות שני בתים (Bytes) ולא באמצעות בית בודד, כפי שהיה נהוג בשיטה הקודמת. לכן, מסדי נתונים שמבוססים במידה רבה על שדות מבוססי-תו, עלולים להכפיל את גודלם. Jet מתגבר על הבעיה על ידי דחיסת הנתונים ופרישתם מחדש לפי הצורך. הוא דוחס שדות תזכיר (Memo) באורך 4,000 תווים או פחות, כך חלק משורות שדה נתוני התזכיר נדחסות, אם כי לא כולן. Access וגם ADO דוחסים אוטומטית תווים בקידוד Unicode, אך DAO אינו תומך בדחיסת נתונים מסוג מחרוזת. מתכנתים שמפתחים ב- Jet SQL יכולים לבחור באפשרות WITH COMPRESSION בעת הגדרת טבלאות.

נוסף לתבנית הייצוג של סוגי הנתונים, גודל הדף הוכפל ומעתה גודלו 4KB, דבר שמצמצם את הפעולות הבו-זמניות שמקורן בהתנגשויות נעילת דף (page-lock). Jet פותר את הבעיה באמצעות נעילת שורה יחידה. ניתן להקטין את בעיית הבו-זמניות על ידי נעילת רשומות בודדות במקום נעילת דפים שלמים. Access 2000 מאפשר למשתמשים לעדכן שתי רשומות בו-זמנית באותו דף.



הערה:

גודל הדף החדש מגדיל את הגודל המקסימלי של מסד הנתונים מ-1.07 ג"ב עד 2.14 ג"ב.

נעילה ברמת שורה בודדת היא אפשרות ברירת המחדל הכללית, אך משתמשים ומפתחים יכולים לנצל את הנעילה ברמת הדף, כפי שהיה מקובל בגרסאות קודמות. שדות תזכיר ואינדקסים אינם תומכים בנעילה ברמת שורה בודדת. גישה לנתונים באמצעות ADO או Jet SQL מאפשרת לחזור לנעילה המקובלת ברמת הדף. גישה לנתונים באמצעות טפסי Access ו-DAO מתבצעת תמיד באמצעות נעילה ברמת שורה בודדת.

נעילה ברמת שורה בודדת חיונית לצמצום התנגשויות בו-זמניות, אך היא כרוכה בתקורה גבוהה של הגדרה והסרת נעילות רשומה. השפעת התקורה תלויה בגודל הרשומה יחסית לגודל הדף, ולמספר הנעילות הדרושות במשימה אחת.

המעבר ל-Unicode מאפשר לבצע מיון תואם Microsoft Windows NT, מה שמשפר את התאימות והביצועים, מכיון ש- Access 2000 מסוגל לבצע מיון עקבי תחת מערכות ההפעלה Windows 95 ו-Windows NT כאחד. הדבר אפשרי מכיון ש-Windows 95 תומכת במיון בשפת ברירת המחדל של המערכת, בעוד ש-Windows NT תומכת במיון תקין בשפות רבות.

בנוסף, Visual Basic 6 ו- SQL Server 7 תומכות בתקן מיון זהה, וכך ניתן לתקנן מוצרי תוכנה רבים. ברוב השפות Access 2000 מבצע מיון במהירות גדולה ב-50%, אך בשפות מסוימות (כגון תאילנדית) הוא עושה זאת במהירות גדולה אף יותר.

Jet 4 תומך ביצירת אינדקס של 255 התווים הראשונים של נתונים מסוג תזכיר (Memo). גרסאות מוקדמות של Access ו-Jet לא תמכו ביצירת אינדקס של שדות תזכיר. האורך המוגבל של האינדקס אינו עונה על הדרישות של יישומי שדות תזכיר, אך יש לו משמעות מיוחדת עבור נתונים מסוג היפר-קישור, שנגזר מסוג הנתונים תזכיר. שיטת יצירת האינדקס החדשה מסוגלת לשפר את המיון והחיפוש של שדות היפר-קישור.

Jet 4 גם מאפשר למפתחים לציין ערך התחלתי וערך צעד (step value) עבור שדות מספור אוטומטי בהפרשים קבועים (auto-incrementing). בנוסף, המשפט החדש ALTER TABLE מאפשר למפתחי Jet SQL לאפס את ערכי ההתחלה והצעד. כדי לשחזר את הערך החדש של עמודת המספור האוטומטי בוחרים משפט SELECT @@IDENTIFY חדש. יש להעביר את ערך משפט SQL כטקסט SQL המיועד עבור השיטה Open של אובייקט ערכת הרשומות (recordset) במשפט ADO. DAO אינה תומכת במשפט החדש.

Jet 4 כולל את הטכנולוגיה המשופרת הניתנת להתקנה ISAM, בכמה תחומים. ISAM עשויה להשפיע על הזדמנויות הפיתוח ב-Access כשיש להשתמש במקורות נתונים חיצוניים. הגירסה החדשה Text/HTML של ISAM מאפשרת לקרוא מסמכי Web שמבוססים על תבנית Unicode. Exchange ISAM החדש כולל שיפורים בתחומים אחדים. ראשית, הוא קורא אינדקסים מתוך Exchange Server, דבר שמאיץ בצורה דרמטית את תהליכי החיפוש אחר רשומות במקורות נתונים של Exchange. בנוסף, הגירסה החדשה של ISAM תומכת ב**פנקס הכתובות של Windows** (Windows Address Book) של הלקוח Microsoft Outlook Express. בעסקים קטנים שבהם הדואר מיועד בעיקרו למשלוח חיצוני ולא פנימי, יש בכך יתרונות משמעותיים (אם כי בתחום הפריון האישי בלבד). שיפור נוסף הוא יכולת Jet לאחזר לקוחות Outlook מוגדרי-התאמה אישית וגם של לקוח Outlook תחת Exchange server. משתמשי מסדי הנתונים dBase ו-Paradox יכולים להמשיך לגשת למסדי נתונים אלה עם זכויות קריאה וכתיבה באמצעות Jet 5. משתמשים הזקוקים לתמיכה בגירסה עדכנית יותר, חייבים להשיג עותק של Borland Database Engine מיצרן תוכנה אחר.

סוגי נתונים

VBA משתמש במשתנים לשמירת תוצאות חישוב, להגדרת מאפיינים, שליחת ארגומנטים לשיטות ולהעברת ערכים בין שגרות. כדי שיתפקד בצורה יעילה, VBA מבסס משתנים אלה על קבוצת סוגי נתונים. בסעיף זה נתמקד בסוגי נתונים של משתנים. סוגים נוספים של נתונים דומים באופן כללי, אך Access מאפשר להשתמש בסוגי נתונים שונים לכל משימה.

הטבלה שלפניך מציגה את סוגי הנתונים העיקריים ודרישות האחסון שלהם בתוכניות VBA. מפתחים הזקוקים לרמות דיוק גבוהות בחישוביהם המספריים יכולים לנצל לשם כך את סוג הנתונים Decimal (עשרוני) בתור קבוצת-משנה של סוג הנתונים Variant. סוגי הנתונים Decimal הם מספרים שלמים לא חתומים של 12 בתים שקנה המידה שלהם מציין את המספר המרבי של ספרות עשרוניות שניתן לאחסן מימין לנקודה העשרונית. לא ניתן להצהיר על סוג נתונים Decimal באמצעות משפט Dim, אך אפשר לאחסן סוג נתונים כזה על ידי הפיכת Variant בעזרת הפונקציה CDec. כללית, רצוי להשתמש בסוג הנתונים הקטן ביותר האפשרי, כדי להותיר מקום בזיכרון עבור משתנים נוספים ופקודות ביצוע של יישומים. יחד עם זאת, כאשר זקוקים לדיוק גדול בפעולות חישוביות, ניתן להשתמש בסוג הנתונים Decimal.

אם לא תצהיר על משתנה בתור סוג מבין המוגדרים, Access יקצה לו את סוג הנתונים **Variant**. סוג נתונים זה הוא גמיש, מכיון שהוא מטפל בערכים מספריים ובערכי מחרוזת כאחד. סוג הנתונים **Variant** יכול לאחסן את כל שאר סוגי הנתונים חוץ מסוגים מוגדרי-משתמש (שנגזרים מכל שאר הסוגים). כש-VBA מבצע פעולה על שני נתונים מסוג **Variant**, עליו לאחזר תחילה את תוכן הנתון השני ולהמיר את התוצאה לסוג המשנה של ה-**Variant** הראשון. תהליך זה מאט את הטיפול במשתני Variant לעומת משתנים מסוגים אחרים. בנוסף, סוג המשנה של הנתונים המוחזר באמצעות הפעולה, עלול להיות שגוי בהקשר של היישום.

באפשרותך להשתמש בפונקציה VarType כדי לקבוע את סוג המשנה של סוג הנתונים Variant. הפונקציה קולטת נתון מהסוג Variant ומחזירה קבוע VBA, שמציין את סוג המשנה של הערך בתור סוג נתונים Variant. לדוגמה, varMyVariable מכיל את הערך "Hi, there!", הפונקציה VarType(varMyVariable) מחזירה את vbString(8). באופן דומה, אם varMyVariable מכיל את התאריך #1/1/2000#, הפונקציה VarType(varMyVariable) תחזיר את vbString(7). עיין בתיעוד המקוון או ב-Object Browser (סורק האובייקטים) לקבלת רשימה מלאה של קבועי סוג-משנה של Variant המוחזרים על ידי VarType.

פונקציות ההמרה של VBA מאפשרות להמיר סוג נתונים Variant לסוג-משנה מוגדר, וגם לציין את סוג התוצאה לה מצפים. לדוגמה, CDbl(varMyVariable) מחזירה את תוכן המשתנה varMyVariable בתור משתנה מהסוג Double.

על אף הסיכוי לקבלת טעות והצורך בעיבוד נוסף, Variant הוא סוג נתונים מקובל. סוג ברירת המחדל של נתוני טבלה, שאילתה, דוח ושדה טופס, הוא Variant. אם תשאיר בטבלה שדה לא מוגדר, הוא יחזיר ערך ריק (Null) ברגע שתפעיל עליו שאילתה (בהנחה שלא נשלחו אליו נתונים), שהוא אחד משני ערכי הנתונים המיוחדים של Variant. הערך Null מציין נתון חסר, לא ידוע, או לא ישים. הפונקציה IsNull של VBA בודקת אם הערך שנבחר הוא Null, והאופרטור Is Null בקריטריון שאילתה עושה פעולה זהה בשדה. טפל בזהירות במשתנים המכילים Null, מכיון שערכי Null נוטים להתפשט. צירוף של משתנה ששווה ל-Null עם משתנה כלשהו אחר, יחזיר תמיד ערך Null.

סוגי נתונים של משתנים

שם	מספר בתים	טווח
בית (Byte)	1	0-255
בוליאני (Boolean)	2	True או False
מספר שלם (Integer)	2	-32,768 עד 32,767
ארוך (Long)	4	-2,147,483,648 עד 2,147,483,648
יחיד (Single)	4	-3.402823E38 עד -1.401298E-45 עבור ערכים שליליים; 1.40129E-45 עד 3.402823E38 עבור ערכים חיוביים
כפול (Double)	8	-1.79769313486232E308 עד -14.94065645841247E-324 עבור ערכים שליליים; 4.94065645841247E-324 עד 1.797969313486232E308 עבור ערכים חיוביים
מטבע (Currency)	8	-922,337,203,685,477.5808 עד 922,337,203,685,477.5807
תאריך (Date)	8	1 בינואר, 100 עד 31 בדצמבר 9999
אובייקט (Object)	4	הפניה לאובייקט (ראה משפט Set בעזרה המקוונת).
מחרוזת קבועה (Fixed String)	אורך המחרוזת	עד 64,000 תווים
מחרוזת משתנה (Variable String)	10+אורך המחרוזת	עד כ-2 מיליארד תווים
Variant (עם מספרים)	16	כמו כפול
Variant (עם תווים)	22+אורך המחרוזת	כמו כפול
מוגדר משתמש	תלוי באלמנטים	סך האלמנטים המרכיבים את סוג הנתונים המותאם אישית



הערה:

מאחר שרק נתונים מהסוגים Variant יכולים לקבל ערכי Null, עליך להשתמש בסוג זה בכל מקרה שהיישום שלך יכול לצאת נשכר מכך. ערכי Null מבטלים את הצורך בערכים שרירותיים לציון נתונים חסרים בשדה או בסוג משתנה.

מילת המפתח Empty (ריק) משמשת בתור סוג-משנה ומייצגת ערך משתנה שלא אותחל. ב-VBA משתמשים בפונקציה IsEmpty כדי לקבוע אם משתנה הוא בעל ערך כזה. Empty, Null, 0 ומחרוזת ריקה מתווים ("") שונים זה מזה. כש-VBA הופך משתנה מסוג Variant שערכו Empty, הוא הופך את המשתנה ל-0 או ל-"", בהתאם לסוג הערך ההולם ביותר במקרה זה, מספר או מחרוזת.

כשעליך לקרוא לפונקציות API של Windows באמצעות ספריות קישור דינמי (DLL), לעיתים תזדקק למשתנה מוגדר-משתמש. עליך להצהיר על משתנים מוגדרי משתמש בין המשפטים Type ו- End Type. השורות הבאות בין שני משפטים אלה צריכות להגדיר את האלמנטים של סוג המשתנה המותאם אישית. למשל, אפשר להתייחס לספר בתור אוסף אלמנטים, כולל ISBN (מספר ספר סטנדרטי בינלאומי), שם הספר, שם המחבר (או המחברים), המו"ל ומספר דפי הספר. באפשרותך להגדיר את סוג הנתונים המותאם אישית כך:

```
Type Book
    ISBN as Long
    Title as String
    Authors as String
    Publisher as String
    Pages as Integer
End Type

Sub MyBook
Dim udvMyBook as Book
    udvMyBook.ISBN = 1234567890
    udvMyBook.Title = "Programming Microsoft Access 2000"
    udvMyBook.Authors = "Rick Dobson"
    udvMyBook.Publisher = "Microsoft Press"
    udvMyBook.Pages = 550
End Sub
```

מוסיפים את זוג המשפטים Type ו-End Type באזור ההצהרות הכללי במודול. השיגרה MyBook יוצרת מופע (instance) של משתנה מוגדר-משתמש שנקרא Book.

הצהרות

ניתן להצהיר על משתנים וקבועים (**קבוע** הוא משתנה מיוחד שלא ניתן לשנות את ערכו לאחר שנקבע במשפט Constant). להצהרה שני תפקידים: היא קובעת את **טווח ההכרה** (scope) של המשתנה (התחום שבו היישום מכיר את המשתנה), ומגדירה את סוג הנתונים של המשתנה. אלו שתי סיבות טובות להצהיר על משתנים בטרם מתחילים לטפל בהם. באמצעות המשפט Option Explicit שבאזור General (כללי) של המודול תוכל לדרוש הצהרה על משתנה.

המשפט Public (ציבורי) באזור Declaration (הצהרות) של מודול משמש כדי להצהיר על משתנים שהיישום יכול לטפל בהם מתוך שיגרה כלשהי שנמצאת ביישום, כולל אלו שנמצאות במודולים אחרים. באפשרותך להשתמש במשפט Private (פרטי) כדי להגדיר במפורש משתנה כבעל טווח הכרה מקומי במודול (הדבר אינו הכרחי, כיון שהמשפטים Dim ו-Static מצהירים על משתנים שהם פרטיים במודול, כברירת מחדל).

ההצהרה Dim שומרת מקום בזיכרון עבור משתנה רק עד סיום השיגרה שבה מופיעה ההצהרה על המשתנה. משמעות הדבר, שמשתנים שהוצהרו באמצעות משפט Dim מאבדים את ערכם לאחר הקריאה לשיגרה, עד הקריאה הבאה. משתנים שהוצהרו בעזרת משפט Static נשמרים לאורך חיי המודול או עד אתחול היישום או הפעלתו מחדש (אחד השימושים של הצהרה על משתנים סטטיים הוא עריכת סיכומים או קביעת מספר הפעמים שהשיגרה הופעלה).



הערה:

באפשרותך לנקות את הערכים מהמשתנים הסטטיים בשיגרה על ידי בחירה באפשרות **Reset** מתפריט **Run** של VBE.

ניתן להשתמש במילת המפתח Static בהצהרות על פונקציות ושגרות וגם בהצהרות על משתנים. מילת המפתח פועלת בצורה זהה עבור פונקציות ושיגרות, בכך שהיא משמרת את כל המשתנים המקומיים בפונקציה או בשיגרה, לכל אורך חיי המודול.

מוסיפים את מילת המפתח As להצהרה Dim או Static, כדי לציין סוג נתונים של משתנה. המשפט Dim inMyNumber As Integer מצהיר על סוג הנתונים של המשתנה inMyNumber בתור מספר שלם.

מערכים

Access מאפשר להצהיר על **מערכים** (arrays). מערכים הם משתנים המכילים רשימות ערכים מאותו סוג. מערך יכול להיות רב-מימדי, ומספר המימדים שלו עשוי להגיע ל-60. לדוגמה, המשפט Dim aryScores(2, 1) As Integer, מקצה מערך דו-מימדי של שישה אלמנטים: שלוש שורות ושתי עמודות של מספרים שלמים (הספירה מתחילה מ-0).

כברירת מחדל, מערכים הם מבוססי-אפס. המשפט Dim aryMyName(1) כולל שני אלמנטים, שהקוד יכול להפנות אליהם בתור aryMyName(0) ו-aryMyName(1). השתמש במשפט Option Base 1 באזור Declarations כדי לציין מערך כמבוסס-אחד. השיגרה שלפניך מצהירה על מערך בן שני אלמנטים, מגדירה את ערכיו ומדפיסה את האלמנטים שלו ואת סכום האלמנטים בחלון Immediate.

```
Sub ArrayTest()
Dim aryMyExpenses(1) As Currency, Total As Currency
   aryMyExpenses(0) = 10.5
   aryMyExpenses(1) = 22.25
   Total = aryMyExpenses(0) + aryMyExpenses(1)
   Debug.Print aryMyExpenses(0) & " + " & aryMyExpenses(1) & " = " & Total
End Sub
```

המערך aryMyExpenses מכיל שני אלמנטים שסוג הנתונים שלהם הוא Currency (מטבע). סוג נתונים זה כולל שמונה בתים, ולכן המערך aryMyExpenses כולל 16 בתים. מאחר שבאפשרותך להגדיר מערכים בעלי מספר רב של מימדים ומחרוזות באורך משתנה, משאבי הזיכרון עלולים להתכלות במהירות.

אתה משתמש במילות המפתח Dim, Static, Public ו-Private להגדרת טווח ההכרה של מערך, כמו עבור משתנים סקלאריים. מילת המפתח ReDim משמשת לשינוי דינמי של מימדים במערך רב-מימדי.

לוגיקה של התניות ומבני לולאה

ביצוע מותנה של קוד ולולאות מהווה מרכיב בסיסי בפתרונות מבוססי-קוד רבים. VBA כולל מבחר אפשרויות גדול ליישום תהליכים אלה. הסעיפים הבאים סוקרים את סוגי המשפטים העיקריים המאפשרים ביצוע מותנה של תוכנית וכוללים דוגמאות מעשיות.

If...Then

שגרות רבות אינן ממלאות את תפקידן על ידי ביצוע סדרתי של שורות הקוד. לעיתים רוצים להתנות את ביצוע הקוד – לדלג על שורות מסוימות ולבצע שורות אחרות. אחת הדרכים הגמישות והבטוחות לעשות זאת בשגרות VBA היא בעזרת משפטי If...Then. משפט בסיסי זה כולל למעשה שלוש וריאציות בסיסיות. הראשונה היא ביצוע מותנה של קטע קוד פשוט. כך נראה התחביר שלה:

```
If condition Then
   ' Statements
End If
```

קטע מהסוג If...End יכול להכיל בתוכו משפט אחד או יותר. המשפט End If מציין את סוף הקטע. VBA מבצע את המשפטים שבתוך הקטע, רק כאשר תנאי מסוים מתממש (כלומר, מקבל ערך True). ניתן לקנן קטעי If...End רבים זה בתוך זה.

הווריאציה השנייה של משפט If...Then מאפשרת לקוד לבצע אחד מתוך שני קטעי קוד. כך נראה התחביר שלה:

```
If condition Then
   Statements1
Else
   Statements2
End If
```

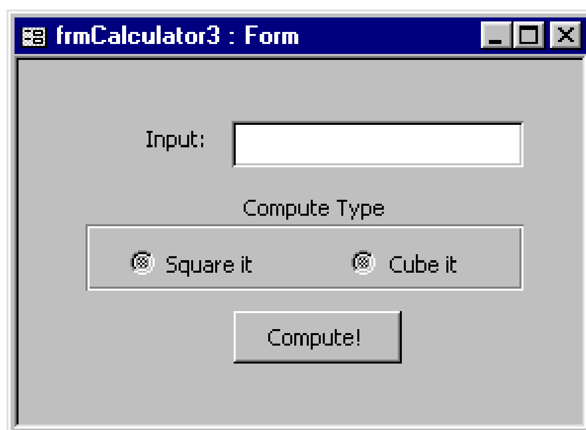
סוג משפט זה מבצע קטע קוד אחד מתוך שניים. כשמתמלא תנאי הביצוע, מתבצעים משפטי הקבוצה הראשונה. אחרת, מתבצעים משפטי הקטע השני.

עיצוב זה גמיש יותר, אך הוא עדיין מגביל אותנו לשתי אפשרויות בלבד. לאמיתו של דבר, באפשרותך לקנן משפטי If...Then כך שמספר האפשרויות יגדל, אך VBA כולל דרך נוספת שמפשטת ביצוע מותנה של קטע קוד אחד או יותר, עם התחביר הבא:

```
If condition1 Then
   ' Statements
ElseIf condition2 Then
   ' Statements
Else
   ' Statements
End If
```

צורה זו של משפט If...Then משלבת תנאים רבים ושלוש קבוצות משפטים, או יותר. באפשרותך להוסיף קבוצות משפטים ותנאים חדשים על ידי הוספת משפטי ElseIf ובהם התניות ומשפטים משלהם. דרך זו מציעה אפשרויות רבות יותר מקודמתה, לא זו בלבד שהיא מאפשרת מספר גדול יותר של תנאים, אלא שהיא גם מגבילה את הביצוע של כל קטע קוד (למעט האחרון) למקרה שבו תנאי הבדיקה מתמלא במלואו. בקטע If...End הקודם, משפט If...Then השני מבצע את קטע המשפטים השני כל פעם שהתנאי שבמשפט הראשון אינו מתמלא. כאשר אי-מילוי התנאי הראשון אינו מהווה אוטומטית סיבה לבצע את קטע הקוד השני, יש צורך בתבנית התניה נוספת.

עיין בטופס שבתרשים 1.15. הטופס כולל תיבת טקסט, קבוצת לחצני אפשרויות ולחצן פקודה. קבוצת לחצני האפשרויות מאפשרת למשתמש להעלות את הערך שבתיבת הטקסט בריבוע או בחזקת 3. כדי לחשב ריבוע של מספר, יש להקליד את המספר בתיבת הטקסט, לבחור באפשרות **Square it** ולבסוף ללחוץ על לחצן **Compute!**.



תרשים 1.15: טופס לחישוב חזקות ריבועיות או חזקות שלישיות של מספרים

הקוד שלפניך מציג את שתי השגרות שמאפשרות לבצע העלאה בריבוע. שגרת האירוע cmdComputer_Click מגיבה ללחיצה. אם קבוצת לחצני האפשרויות (opgComputerType) מקבלת את הערך 1, סימן שהמשתמש בחר באפשרות Square it (לחצן האפשרות העלאה בריבוע). אם קבוצת לחצני האפשרויות שווה ל-1, השיגרה קוראת לשיגרה MySquarer. אחרת, היא מסתיימת מבלי לבצע חישובים כלשהם.

```
Sub cmdComputer_Click()

    If opgComputeType = 1 Then
        MySquarer txtInput.Value
    End If

End Sub

Sub MySquarer(MyOtherNumber As Double)

    dblResult = MyOtherNumber * MyOtherNumber
    MsgBox dblResult, vbInformation, "Programming Microsoft Access 2000"

End Sub
```

השיגרה cmdComputer_Click מנצלת את מבנה If...Then הראשון. פעולתה העיקרית היא ביצוע מותנה של קטע קוד אחד. במקרה זה, הקטע מכיל שורה אחת בלבד. אם המשתמש בוחר באפשרות Square it, MySquarer מכפילה את הערך שבתיבת הטקסט בעצמו, ומציגה את התוצאה בתיבת הודעה.

אם המשתמש בוחר באפשרות Cube It, בטרם לחץ על לחצן הפקודה, הקוד לא ייבצע פעולה כלשהי. קבוצת האפשרויות מחזירה את הערך 2, אך הקוד אינו כולל תנאי לזיהוי מצב זה. לאמיתו של דבר, הקוד אינו מכיר קבוצת אפשרויות אחרת מ-1. הקוד הבא מציג גירסה חדשה של שגרת אירוע יחד עם הקוד שמטפל במצב בו המשתמש אינו בוחר באפשרות Square it.

```
Sub cmdComputer_Click()
    If opgComputeType = 1 Then
        MySquarer txtInput.Value
    Else
        MyCuber txtInput.Value
    End If
End Sub

Sub MyCuber(MyOtherNumber As Double)
Dim dblResult As Double
    dblResult = MyOtherNumber ^ 3
    MsgBox dblResult, vbInformation, "Programming Microsoft Access 2000"
End Sub
```

השיגרה cmdComputer_Click מנצלת את התבנית השנייה של משפט If...Then. היא קוראת לפונקציה MySquarer אם opgComputerType שווה ל-1, אך בכל מקרה אחר היא קוראת ל-MyCuber. הקוד פועל בצורה תקינה כשהמשתמש בוחר באפשרות Cube It. לעומת זאת, נותרה לפחות בעיה אחת: הטופס נפתח מבלי שאפשרות פעולה כלשהי נבחרה מראש.

אם המשתמש מקליד ערך בתיבה ולאחר מכן לוחץ על לחצן הפקודה, הטופס יחזיר את החזקה השלישית של הערך על אף העובדה שהמשתמש לא בחר באף אחת מהאפשרויות. הבעיה נעוצה בתכנון המשפט If...Then. אנו זקוקים לווריאציה השלישית עם שני תנאים נפרדים: אחד להעלאה בריבוע והאחר להעלאה בחזקת שלוש. הקוד שלפניך מדגים זאת:

```
Sub cmdComputer_Click()
    If opgComputeType = 1 Then
        MySquarer txtInput.Value
    Else if opgComputeType = 2 Then
        MyCuber txtInput.Value
    Else
        msgbox "Please select computing option"
    End If
End Sub
```

דוגמה זו כוללת שני תנאים. האחד בודק אם קבוצת האפשרויות שווה ל-1, והאחר בודק אם היא שווה ל-2. אם תוצאות הבדיקה אינן עונות על אף אחד מהתנאים, השיגרה תציג תיבת הודעה ובה הוראה למשתמש לבחור את סוג החישוב. קל ופשוט יהיה עתה להוסיף לחצני אפשרות חדשים לקבוצת האפשרויות ועל ידי כך להגדיל את מיגוון החישובים. כל שעליך לעשות הוא להוסיף משפט ElseIf חדש ובו תנאי מיוחד עבור כל לחצן שהוספת.

כפי שתוכל לראות, המשפט If...Then מצטיין בגמישותו. באפשרותך לנצל אותו לטיפול באפשרויות בחירה רבות, אך הוא מטפל בפריט אחד או שניים ביתר הצלחה. תחביר המשפט אף הוא משתנה קמעה בהתאם למשימה שברצונך לבצע.

Select Case

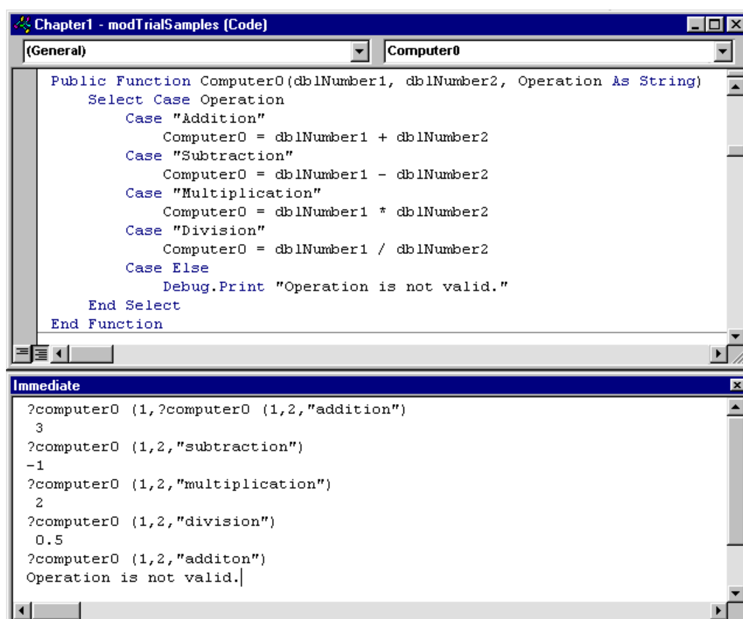
המשפט Select Case מבצע פעולה אחת: הערכת ביטוי וביצוע מותנה של קטע קוד אחד. כשמטפלים ביותר מאפשרות מותנית אחת או שתיים, קל יותר להגדיר ולתחזק משפט Select Case לעומת אוסף משפטי If...Then. התחביר הכללי של Select Case הוא:

```
Select Case test expression
   Case expression list-1
      ' Statements
   Case expression list-2
      ' Statements
   Case Else
      ' Statements
End Select
```

משפט Select Case מעריך ביטוי בדיקה התחלתי. הביטוי יכול להיות פשוט כמו פרמטר שהועבר, או מורכב כמו ביטוי המשמש בהנדסת טילים. expression list-1 ו- expression list-2 הם טווחים (או ערכים מוגדרים) עבור ביטוי הבדיקה. הגדרת רשימות ביטויים יכולה לכלול, קבועים, סדרת פריטים מופרדים, או ביטויי אמת או שקר. כאשר תוצאת רשימת ביטויים היא אמת מכיון שהיא תואמת לביטוי הבדיקה, יתבצע קטע משפטים מתאים. דוגמת התחביר שהוצגה מציגה שתי רשימות בלבד, אך תוכל להגדילה על ידי שילוב משפטי Case נוספים. משפט Case Else הוא אופציונלי, אך הוספתו תורמת לעמידה בכללי התכנות התקני. אפשרות זו לוכדת ערכי ביטוי בדיקה שלא נלכדו על ידי משפטי Case קודמים. ניתן לקנן משפטי Select Case זה בזה; כל מופע חייב להתחיל במשפט Select Case ולהסתיים במשפט End Case.

תרשים 1.16 מציג משפט Select Case המשמש בשגרת פונקציה. שיגרה זו מטפלת בזוג מספרים שהועברו אליה באמצעות אחת מארבע פעולות מספריות. ארגומנט מחרוזת מציין את הפעולה, והחלון **Immediate** (מיידי) מציג את תוצאת הפעולה שנבחרה. כפי שניתן לראות בחלון המיידי, הבדיקות אינן רגישות לאותיות רישיות.

פעולה שאויתה בצורה שגויה גורמת לתוכנית להגיע למשפט Case Else. במקרה זה, הפונקציה תדפיס הודעה בחלון המיידי, בה ייאמר כי הפעולה שנבחרה אינה חוקית.


```
Public Function Computer0(dblNumber1, dblNumber2, Operation As String)
    Select Case Operation
        Case "Addition"
            Computer0 = dblNumber1 + dblNumber2
        Case "Subtraction"
            Computer0 = dblNumber1 - dblNumber2
        Case "Multiplication"
            Computer0 = dblNumber1 * dblNumber2
        Case "Division"
            Computer0 = dblNumber1 / dblNumber2
        Case Else
            Debug.Print "Operation is not valid."
    End Select
End Function
```

```
?computer0 (1,?computer0 (1,2,"addition")
 3
?computer0 (1,2,"subtraction")
-1
?computer0 (1,2,"multiplication")
 2
?computer0 (1,2,"division")
 0.5
?computer0 (1,2,"additon")
Operation is not valid.
```

תרשים 1.16: שגרת פונקציה והפלט שלה ממחישים את השימוש במשפט Select Case

For...Next

המשפט For...Next משמש בעיקר לביצוע קטע קוד בצורה חוזרת (לולאה), מספר פעמים ידוע. ניתן לצאת מתוך הלולאה באמצעות המשפט Exit For, ולכן ניתן לנצל את המשפט For...Next כשלא יודעים בדיוק כמה פעמים יש לבצע את הקוד. אך יש לציין את מספר הלולאות המקסימלי. התחביר הכללי של משפט For...Next הוא:

```
For counter = start To stop Step step
    ' Statements
    If condition Then
        Exit For
    End If
    ' Statements
Next counter
```

לולאת For מתחילה בשורה For counter ומסתיימת בשורה Next counter. המשפט For...Next מאתחל את counter לערך של start ולאחר מכן מבצע את המשפטים שבין For ו-Next. מהלך הביצוע חוזר תמיד לראש הלולאה, כש-counter גדל בערך האופציונלי של step (ערך ברירת המחדל של step הוא 1).

בדוגמה, משפטי לולאת For מבוצעים פעם אחר פעם, עד שערכו של counter עובר את זה של stop (או כאשר counter קטן מ-stop במקרה ש-step שלילי). בשלב זה השליטה עוברת למשפט הבא מייד לאחר Next counter. המשפט האופציונלי Exit For גורם ללולאה להסתיים לפני ש-counter עובר את הערך של stop.

ניתן לקנן משפטי For...Next זה בזה. לולאה פנימית מעבירה את השליטה ללולאה חיצונית כשהערך של counter עובר את זה של stop. VBA מסוגל לגרום לשגיאת זמן ריצה, אם הוא נתקל במשפט Next ללא בן זוגו For, אך סביר יותר שלפני כן תקבל שגיאת תחביר.

השיגרה CountFor הבאה מיישמת את הלוגיקה של לולאת For...Next ובו-בזמן מבססת את טכניקות העבודה עם מערכים והצהרות Static (המערך aryMyArray כולל חמישה אלמנטים. כזכור, אינדקס של מערך מתחיל ב-0, אלא אם כן הקוד מציין אחרת במפורש). הקוד מכיל הצהרה על משתנה מסוג Integer שיישמש כמונה. המשפט For...Next מציב באופן עוקב את הערכים 1-5 במשתנה המונה, intIndex.

שני משפטים מתבצעים לכל מעבר בלולאה. ראשית, ערך אלמנט במערך aryMyArray מחבר את הערך הנוכחי של intIndex לערך הנוכחי שלו. הקוד מצהיר על aryMyArray באמצעות משפט Static, ולכן האלמנטים של המערך שומרים על ערכיהם גם כשהשיגרה מתבצעת פעמים רבות. לאחר הביצוע הראשון של השיגרה, האלמנטים במערך זהים לערכים העוקבים של intIndex. לאחר הביצוע הבא, ערכי האלמנטים גדולים פי שניים מאלה של intIndex וכך הלאה בכל ביצוע מוצלח של השיגרה. המשפט השני מדפיס את הערך הנוכחי של intIndex והאלמנט הקשור אליו מתוך המערך. זכור כי באפשרותך לאתחל מחדש את ערכי האלמנטים במערך ל-0, על ידי בחירת האפשרות **Reset** מתוך תפריט **Run** של VBE.

```
Sub CountFor()
Static aryMyArray(4) As Integer
Dim intIndex As Integer
    For intIndex = 1 To 5
        aryMyArray(intIndex - 1) = _
            aryMyArray(intIndex - 1) + intIndex
        Debug.Print intIndex, aryMyArray(intIndex - 1)
    Next intIndex
    Debug.Print vbLf
End Sub
```

תרשים 1.17 מציג את פלט החלון **Immediate** בשלושה ביצועים עוקבים של השיגרה. העמודה הראשונה מציגה את הערכים העוקבים של intIndex, והשנייה מציגה את הערכים המתאימים של האלמנטים במערך. בביצוע השיגרה הראשון, האלמנטים של המערך תואמים בדיוק את אלה של intIndex. בביצועים השני והשלישי, האלמנטים גדולים פי שניים ופי שלושה בהתאמה. תוצאת הסכימה ממחישה את השפעת ההצהרה על משתנה כ- Static. אם תשנה את מילת המפתח Static ל-Dim, ערכי intIndex וערכי aryMyArray יתאימו בדיוק אלה לאלה בכל ביצועי השיגרה.

```
Immediate
1        1
2        2
3        3
4        4
5        5

1        2
2        4
3        6
4        8
5        10

1        3
2        6
3        9
4        12
5        15
```

תרשים 1.17: החלון **Immediate** מציג את תוצאות ביצוע השיגרה CountFor שלוש פעמים רצופות

For Each...Next ו- With...End With

המשפטים For Each...Next ו- With...End With יכולים להשתלב יחד היטב בפעולות אימות נתונים ומשימות ספירה מקובלות. For Each...Next מבצע הליך חוזר דרך אוסף כלשהו (כגון פקדים בטופס) או מערך. אין דרישה לדעת את מספר האלמנטים באוסף או במערך. המשפט With...End With יכול להשלים את For Each...Next על ידי פישוט אופן הקידוד של פקדים רבים בקטע קוד.

המשפט For Each...Next דומה וגם מורכב פחות מהמשפט For...Next. הדמיון בין השניים נובע מכך שבשניהם הלולאה פותחת בשורה For ומסתיימת בשורה Next. יחד עם זאת, עיצוב השורות For ו-Next שונה בשתי גרסאות לולאת For. המשפט For Each...Next פשוט יותר מהמשפט For...Next, מכיון שהוא פוטר אותך מהצורך לעקוב אחר שלושה פרמטרים נפרדים (counter, start ו-stop) או מההתייחסות לערך חיובי או שלילי של step. המשפט For Each...Next מתחיל תמיד בתחילת האוסף או המערך, ומתבצע בלולאה עד הגיעו לאלמנט האחרון. תחביר המשפט נראה כך:

```
For Each element In group
   ' Statements
   If condition Then
      Exit For
   End If
   ' Statements
Next element
```

המונח group שבשורה הראשונה של המשפט For Each...Next מתייחס לשם האוסף או המערך. element בשורה הראשונה ובשורה האחרונה מציין אובייקטים בודדים באוסף או אלמנטים במערך.

המשפט For Each...Next מבצע בצורה חוזרת את משפטי גוף הקוד על כל אלמנט באוסף או במערך שעליהם הוא פועל. לעיתים תרצה לשלב את המשפט Exit For או משפט אחר שביצוע מותנה, במקום כלשהו בגוף הלולאה For. כך תאפשר לקוד להגיב בצורה דינמית לאירוע מיוחד בסביבת המחשוב. בדיקת התנאי מזהה את האירוע המיוחד, ו- Exit For או משפטים אחרים שביצועם מותנה, יחולו רק בכפוף להתרחשות האירועים.

בדומה למשפט For...Next, ניתן לקנן משפטי For Each...Next זה בזה. בסיום לולאת For Each...Next, השליטה עוברת למשפט הראשון שבא לאחר הלולאה.

המשפט With...End With מפשט את ההתייחסות לשיטות או מאפיינים שונים של אותו אובייקט. בתחילת השורה With ציין את האובייקט שברצונך להתייחס למאפייניו או שיטותיו, וסגור את ההפניה לאובייקט בשורה End With שבסוף הקטע. בין השורות With ו- End With אפשר לגשת למאפיינים או לשיטות האובייקט, מבלי לציין את שמו. הקוד שלפניך מציג את התחביר הכללי של המשפט With...End With:

```
With object
    .propertyname1 = "new value1"
    .propertyname2 = "new value 2"
    .method1
    .method2
End With
```

object הוא שם האובייקט, הפניה לאובייקט או שם מערך. propertyname1 ו-propertyname2 הם מאפייני האובייקט ו-method1 ו-method2 הן שיטות האובייקט. כפי שניתן לראות, המשפט With...End With מאפשר לגשת למאפיינים ולשיטות האובייקט.

תרשימים 1.18, 1.19 ו-1.20 מציגים טופס אימות נתונים בפעולה. למרות שהטופס מכיל רק זוג תיבות טקסט הדורשות אימות, קוד הטופס מנצל לולאת For Each...Next שניתן להרחיבה כך שתכיל פקדי תיבת טקסט נוספים. באפשרותך לערוך שינוי קל ולכלול סוגי פקדים נוספים בשגרת האימות. תרשים 1.18 מציג עיצוב של טופס קליטה בסיסי הכולל זוג תיבות טקסט ולחצן פקודה.

לחיצה על לחצן הפקודה Do It!, מפעילה שגרת אירוע שבודקת את תיבות הטקסט כדי להבטיח שאינן מכילות ערכי Null. אם פקד כלשהו מכיל ערך Null, שגרת האירוע תציג תיבת הודעה שתזכיר למשתמש להזין נתונים בשתי תיבות הטקסט.

שגרת האירוע קוראת גם לשיגרה שמשנה את הרקע של כל תיבת טקסט שמכילה ערך Null מלבן לצהוב. השיגרה גם מעבירה את המיקוד לתיבת הטקסט האחרונה שמכילה ערך Null. הרקע נשאר צהוב, עד שהמשתמש מעדכן את הנתונים בתיבת הטקסט.

frmForEach : Form

Input Form

Input 1:

Input 2: 2

Do It!



תרשים 1.18: תוצאת הלחיצה על לחצן הפקודה Do It! כאשר תיבת טקסט אחת לפחות מכילה ערך Null

תרשים 1.19 מציג תיבת טקסט שהכילה קודם ערך Null, אך כעת מכילה 1. צבע הרקע של תיבת הטקסט משתנה חזרה ללבן ברגע שהמשתמש מזין בה נתונים כלשהם ומעביר את המיקוד מתיבת הטקסט למקום אחר. תרשים 1.20 מציג את הטופס לאחר שהערך החדש מעדכן את תוכן תיבת הטקסט.

frmForEach : Form

Input Form

Input 1: 1

Input 2: 2

Do It!

תרשים 1.19: תיבת טקסט מסומנת בצהוב שהכילה קודם לכן ערך Null, ועכשיו מכילה 1

זוג השגרות הבאות – cmdSubmit_Click ו-MarkFieldsToEdit – בוחן את תיבות הטקסט ומאיר בצהוב תיבת טקסט כלשהי שמכילה Null. שגרת האירוע Click של לחצן הפקודה עוברת בלולאה על כל פקדי הטופס – בין שהם פקדי תיבות טקסט וכאלה שאינם. שגרת האירוע מנצלת את מילת המפתח TypeOf כדי לזהות פקדי תיבת טקסט בין הפקדים. אם לא תעשה זאת, תיגרם שגיאת זמן ריצה, מכיון שלא לכל הפקדים יש מאפיין Value. אם שגרת האירוע מזהה פקד תיבת רשימה, היא מתשאלת את המאפיין Value של הפקד כדי לקבוע אם הוא מכיל ערך Null. פקד שמכיל ערך Null יוזם את הפעלת הקוד הכלול במשפט If...Then. קוד זה מציג את תיבת ההודעה וקורא לשיגרה להאיר את הפקד שחסר לו ערך.

תרשים 1.20: לאחר שהמשתמש מזין מידע ומעביר את המיקוד, צבע הרקע של תיבת הטקסט חוזר להיות לבן

```
Private Sub cmdSubmit_Click()
' Check for valid entries.

    For Each ctl In Screen.ActiveForm.Controls
        If TypeOf ctl Is TextBox Then
            If IsNull(ctl.Value) Then
                MsgBox "Please enter information in both input boxes.", _
                    vbInformation, "Programming Microsoft Access 2000"
                MarkFieldsToEdit
                Exit For
            End If
        End If
    Next ctl

End Sub

Public Sub MarkFieldsToEdit()
    For Each ctl In Screen.ActiveForm.Controls
        If TypeOf ctl Is TextBox Then
            If IsNull(ctl.Value) Then
                With ctl
                    .BackColor = RGB(255, 255, 0)
                    .SetFocus
                End With
            End If
        End If
    Next ctl
End Sub
```

גם השיגרה MarkFieldsToEdit מנצלת את מילת המפתח TypeOf כדי לזהות תיבות טקסט. כשהיא מזהה תיבת טקסט שמכילה ערך Null, היא משתמשת במשפט With...End With כדי לשנות את צבע הרקע של הפקד ומעבירה את המיקוד אל הפקד. על ידי כך היא מבטיחה שתיבת הטקסט האחרונה שמכילה ערך Null תהיה במוקד בתום השיגרה.

כל אחת משגרות האירוע בקוד שלפניך מופעלת בתגובה לאירוע AfterUpdate (לאחר עדכון). כל שיגרה מנצלת קטע מסוג With...End With כדי לשנות את צבע הרקע של הפקד המטופל ללבן, אם צבע הרקע הנוכחי שלו צהוב. האירוע AfterUpdate מתרחש באופן עצמאי עבור שתי תיבות הטקסט, אך הקוד שבכל אחת מהשגרות זהה, למעט שם האובייקט המטופל (txtInput1 או txtInput2).

```
Private Sub txtInput1_AfterUpdate()

   With txtInput1
      If .BackColor = RGB(255, 255, 0) Then
         .BackColor = RGB(255, 255, 255)
      End If
   End With

End Sub

Private Sub txtInput2_AfterUpdate()

   With txtInput2
      If .BackColor = RGB(255, 255, 0) Then
         .BackColor = RGB(255, 255, 255)
      End If
   End With

End Sub
```

Do...Loop

המשפט Do...Loop הוא סוג נוסף של משפט לולאה של יישומי VBA. משפט זה מהווה חלופה גמישה יותר של משפט הלולאה While...Wend – הוא כולל את כל הרכיבים התפקודיים של While...Wend ומוסיף עליהם. VBA ממשיך לתמוך במשפט הלולאה While...Wend משיקולי תאימות לגרסאות קודמות.

באפשרותך לנצל את לולאת Do כדי לבצע בצורה חוזרת קבוצת משפטים עד שתנאי מוגדר יתמלא (כלומר, יקבל ערך True או False). תחביר המשפט Do...Loop תומך במפורש בביצוע בדיקת התנאי טרם ביצוע קטע קוד או מייד לאחר מכן. בדומה למשפטי לולאה אחרים, קיים גם משפט מיוחד ליציאה מקוד הלולאה במהלך הביצוע. לפניך שתי גרסאות תחביר של משפט Do...Loop.

```
Do {While | Until} condition
    ' Statements

    If condition Then
        Exit Do
    End If

    ' Statements

Loop
```

הדוגמה השנייה:

```
Do
    ' Statements

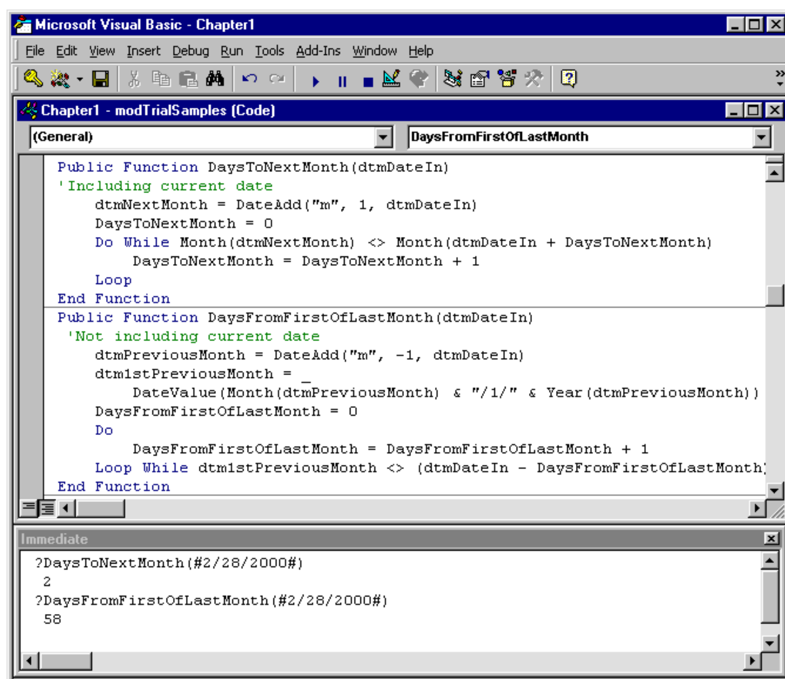
    If condition Then
        Exit Do
    End If

    ' Statements

Loop {While | Until} condition
```

הגירסה הראשונה בודקת את התנאי טרם ביצוע משפטי הלולאה, והשנייה עושה זאת לאחר מכן. בדיקת התנאי יכולה לנצל את מילת המפתח While, או את מילת המפתח Until. מילת המפתח While גורמת ללולאה לשוב ולהתבצע כל עוד התנאי מתקיים (אמת); מילת המפתח Until גורמת ללולאה להתבצע כל עוד התנאי אינו מתקיים (שקר). מפתחים מנוסים בסביבת Access כתבו בוודאי לולאות Do רבות לניווט ברשומות בערכת רשומות עד זיהוי הסימן EOF (סוף קובץ) או BOF (תחילת קובץ). כמו במשפטי לולאה אחרים של VBA, גם כאן ניתן לצאת מלולאת Do לפני תום המחזור על ידי שימוש במשפט Exit Do.

תרשים 1.21 מציג זוג שגרות פונקציה ואת הפלט שלהן, כדי להמחיש תכנות לולאות Do. הפונקציה DaysToNextMonth קולטת תאריך ומחזירה את מספר הימים מאותו תאריך עד ליום הראשון בחודש הבא אחריו. אם תקליד את התאריך הנוכחי, השיגרה תמנה אותו כאחד הימים בתוך מספר הימים הכולל עד החודש הבא. כך, לדוגמה, הפרש הימים בין התאריך 28/2/2000 לבין היום הראשון בחודש שאחריו הוא יומיים (התאריכים 28/2/2000 ו-29/2/2000). הפונקציה פותחת בחישוב dtmnNextMonth, שהוא תאריך הקלט בתוספת חודש אחד (בדוגמה שבתרשים 1.21, התאריך הוא 28/3/2000). אחר כך השיגרה משתמשת בשם הפונקציה כמונה שערכו ההתחלתי הוא 0. השורה השלישית מפעילה לולאת Do שהתנאי שלה בודק את חוסר השוויון בין החודש הבא לבין החודש המתקבל מתאריך הקלט בתוספת המונה. כל עוד אין שוויון, הלולאה מבצעת את קטע הקוד שכולל משפט אחד – הגדלת ערך הפונקציה ב-1.



תרשים 1.21: זוג שגרות פונקציה שמשתמשות בבדיקות תנאי בתחילה ובסיום לולאת Do

הפונקציה DaysFromFirstOfLastMonth מראה את השפעת בדיקת התנאי שמתבצעת בסיום לולאת Do. הפלט הוא מספר הימים בין 28/2/2000 לבין היום הראשון של החודש הקודם (1/1/2000), שהוא 58 (28 ימי פברואר ו-30 הימים האחרונים בינואר). פונקציה זו מתחילה בחישוב החודש הקודם עד היום שצוין. לאחר מכן היא מעבירה חודש זה אל הפונקציה DateValue כדי לקבוע את היום הראשון בחודש הקודם. לפני תחילת לולאת Do, הקוד מספק ל-DaysFromFirstOfLastMonth ערך 0. הכניסה ללולאה אינה מותנית בבדיקה כלשהי, ו-DaysFromFirstOfLastMonth מוגדלת ב-1. בדיקת התנאי בשורה Loop מאפשרת ביצוע מחזור לולאה נוסף, כל עוד ההפרש בין תאריך הקלט לבין DaysFromFirstOfLastMonth קטן מהיום הראשון לחודש הקודם. באופן זה הפונקציה מונה 58 יום מתאריך הקלט עד, אך לא כולל, היום הראשון של החודש הקודם.

פונקציות מוכללות

חלק גדול מהדוגמאות שהצגנו כולל ספריה עשירה של פונקציות מוכללות של Access. הפונקציות יכולות לסייע לזרז תהליכי פיתוח של פתרונות מותאמים אישית על ידי צמצום ופישוט הקוד המותאם אישית. באפשרותך לנצל את הפונקציות המוכללות בדיוק כפי שאתה מנצל את הפונקציות שאתה כותב במיוחד בהזדמנויות רבות, כגון שגרות, שאילתות, טפסים ודוחות. אם תלמד להכיר את השימושים האפשריים

בפונקציות מוכללות, תוכל לגלות אופקים חדשים ויצירתיים ליישום פונקציות מותאמות אישית שתכתוב בעצמך.

סעיף זה כולל שלוש דוגמאות המשתמשות בפונקציות מוכללות: אחת לשאילתה, אחת לטופס ואחת לדוח. באפשרותך (וייתכן שתעשה זאת) למזג את כל שלוש הגישות ביישומיך. הדוגמה השלישית מראה כיצד לגרום לפקד בדוח לתפקד בתור מקור עבור קוד VBA במודול של דוח על ידי שימוש בפונקציות מוכללות. כל שלוש הדוגמאות מבוססות על טבלה קטנה המכילה נתונים להפקת חשבוניות, ומדגימות דרכי דיווח על חשבוניות שמועד פרעונן חלף.

תרשים 1.22 מציג את הטבלה המקורית InvoiceDates בצד שמאל, ובצד ימין שאילתה ובה תצוגה של הטבלה. השאילתה בוחרת שתיים מתוך עשר שורות הטבלה InvoiceDates ומחשבת עמודה חדשה שנקראת Past Due Days (התרשים הוא מתאריך 12/9/98).

InvoiceDates : Table

	InvoiceID	InvoiceDate
	1	06/15/1998
	2	07/01/1998
▶	3	08/17/1998
	4	08/18/1998
	5	08/20/1998
	6	09/01/1998
	7	09/08/1998
	8	09/10/1998
	9	09/12/1998
	10	09/14/1998

Record: 3

qryWayPastDue : Select Query

	InvoiceID	InvoiceDate	Past Due Days
	1	06/15/1998	59
	2	07/01/1998	43
▶	(AutoNumber)		

Record: 3 of 3

תרשים 1.22: הטבלה InvoiceDates ותוצאת שאילתה שבוחרת שורות ומחשבת שדות חדשים בעזרת פונקציות מוכללות

באפשרותך למצוא את היום הנוכחי באמצעות פונקציה מוכללת אחת או יותר: הפונקציה Date והפונקציה Now (ערך ההחזרה של Now כולל גם את השעה הנוכחית. אם ברצונך לטפל בתאריכים בלבד, אין זה משנה באיזו פונקציה תשתמש).

השאילתה כוללת את שתי העמודות של הטבלה InvoiceDates. הקריטריון של העמודה InvoiceDates הוא Date()-30>. התאריך הנוכחי בזמן ביצוע השאילתה היה 12/9/98, ולכן חלף מועד פירעונה של חשבונית מתאריך כלשהו שקדם ל- 13/8/98. מכאן שהשאילתה תחזיר חשבוניות מהתאריכים 1/7/98 ו- 15/6/98.

ערכי Past Due Days ראויים לעמודה נפרדת. שדה מחושב זה משתמש בזוג פונקציות מוכללות: DateDiff ו-Now. הפונקציה DateDiff אידיאלית לחישוב ההפרש בין שני תאריכים. באפשרותך לציין את ההפרש בימים, חודשים, רבעונים, שנים ויחידות זמן שונות נוספות. במקרה של העמודה Past Due Days, הפונקציה DateDiff מחזירה את ההפרש בימים. DatesDiff("d",[InvoiceDate]+30,Now()) הוא הביטוי שמחשב את ערך השדה. ביטוי זה קובע את התאריך הנוכחי באמצעות הפונקציה Now (למרות

שהפונקציה Date יכולה לעשות זאת בצורה זהה). המונח [InvoiceDate] מפנה לטבלה המשמשת כמקור הרשומות עבור השאילתה. התוספת 30+ משמשת לציון תאריך התשלום האחרון המותר, מבלי שהדבר ייחושב כאיחור.

השאילתה יכולה להיות נקודת פתיחה להפקת שימושים עסקיים מעשיים, אך יש לה כמה חסרונות. אחד החסרונות הבולטים הוא שתצוגת גיליון הנתונים אינה מציגה את התאריך הנוכחי, מה שעלול לעורר ספק באשר לבסיס החישוב של חשבוניות שמועד פירעונם חלף. אובייקט הטופס frmPastDueEvaluator שבתרשים 1.23 פותר בעיה זו על ידי הצגת הנתונים בתצוגת הטופס הידידותית. בתצוגה זו, שלושה שדות הטופס מספקים את המידע החשוב ביותר. השדה InvoiceDate מציג את יום הכנת החשבונית. השדה הבא אחריו מציג את התאריך הנוכחי. השדה הבא מחשב את מספר הימים שחלפו מהמועד שנקבע לפירעון החשבונית. שדה זה יכיל את הערך 0, אם החשבונית נוצרה לפני פחות מ-30 יום.

frmPastDueEvaluator : Form

Days Past Due Evaluator

InvoiceID: 1
Invoice Date: 06/15/1998
Today's Date: 09/12/1998
Days late: 59

Record: 1 of 10

תרשים 1.23: טופס לחישוב מספר הימים שחלפו ממועד הפירעון של חשבוניות

שני השדות האחרונים בטופס משתמשים בפונקציות מוכללות לחישוב ערכים. השדה Today's Date (תאריך נוכחי) משתמש בפונקציה Date. במקרה זה, Date עדיפה על Now, מכיון ש-Now מציגה את התאריך וגם את השעה, אם אין מעצבים את השדה אחרת. אם תיתן למאפיין Control Source (מקור פקד) של השדה את הערך ()Date=, אינך צריך לעצבו במיוחד. השדה Days Late מנצל פונקציה מסוג IIF (Immediate If). הפונקציה משמשת לכתיבת ביטויים מובנים ומוכרת גם למפתחי גליונות אלקטרוניים. לדוגמה הביטוי שבשדה האחרון בטופס הוא:
IIF(Date()-[txtInvoiceDate]>30,Date()-[txtInvoiceDate]-30,0). הביטוי כולל שלושה ארגומנטים, בדומה לפונקציית If של הגיליון האלקטרוני. הארגומנט הראשון הוא תנאי שיכול לקבל ערך True או False. אם התנאי מתקיים (אמת), פונקציית IIF מחזירה את ערך הארגומנט השני; אם התנאי אינו מתקיים (שקר), פונקציית IIF מחזירה את ערך הארגומנט השלישי. מכאן שאם התאריך שבתיבת הטקסט של תאריך החשבונית, txtInvoiceDate, מוקדם ב-30 יום או יותר מהתאריך הנוכחי, פונקציית IIF תחזיר את מספר הימים שמעבר ל-30 ימי האורכה; אחרת, פונקציית IIF תחזיר את הערך 0.

הדוח שבתרשים 1.24 מציג גישה נוספת לסיכום חשבונות שמועד פרעונם עבר. הדוח מכיל שלוש עמודות. העמודה האחרונה מחשבת את מספר הימים שחלפו ממועד פירעון החשבונית. בכל שורה נתונה של חשבונית, עמודה זו תהיה ריקה אם מועד פירעון החשבונית לא חלף. כותרת הדוח מזכירה למשתמש את תאריך הדוח, וכך ניתן לדעת את התאריך שמגדיר את החשבוניות שמועד פירעונן חלף.

InvoiceDates

30 Day Past Due Report

Today's Date is: 09/12/1998

InvoiceID	Invoice Date	Past Due Days
1	06/15/1998	59
2	07/01/1998	43
3	08/17/1998	
4	08/18/1998	
5	08/20/1998	
6	09/01/1998	
7	09/08/1998	

Page: 1

תרשים 1.24: דוח שמציג בצורה מותנית את מספר הימים שחלפו ממועד פירעון חשבונית. אם החשבונית נכתבה 30 יום או פחות מהמועד הנוכחי, העמודה האחרונה תהיה ריקה

ערך עמודת הדוח האחרונה מחושב באמצעות נוסחה. תוכל להתבסס על שתי הדוגמאות הקודמות כדי לכתוב את הנוסחה. ייחודו של הדוח הוא בכך שהעמודה האחרונה תהיה לפעמים ריקה. שגרת האירוע OnFormat (בעת עיצוב) של המקטע **פירוט** (Detail) של הטופס מאפשרת זאת על ידי הגדרה מותנית, בזמן ריצה, של המאפיין Visible (גלוי) של פקד העמודה האחרונה. לפניך קוד שגרת האירוע.

```
Private Sub Detail_Format(Cancel As Integer, FormatCount As Integer)
Dim ctlD As Control

    For Each ctlD In Me.Detail.Controls

        If ctlD.Name = "txtPastDueDays" Then
            If ctlD.Value >= 0 Then
                ctlD.Visible = True
            Else
                ctlD.Visible = False
            End If
        End If

    Next ctlD

End Sub
```

שגרת האירוע מנצלת לולאת For Each...Next כדי לעבור בלולאה על הפקדים של כל רשומה בחיפוש אחר תיבת הטקסט txtPastDueDays. תיבת טקסט זו מכילה את מספר הימים שחלפו מאז מועד פירעון החשבונית. אם הערך חיובי, מועד הפירעון חלף, והקוד יגדיר את המאפיין Visible של הפקד כאמת. אחרת, מועד פירעון החשבונית לא חלף והמאפיין Visible של הפקד יקבל ערך שקר. אם כן, מספר מצומצם של שורות קוד יכול ליצור עיצוב דוח גמיש בזמן ריצה.

תקציר פונקציות נבחרות

Access 2000 מכיל מעל 170 פונקציות מוכללות המבצעות מיגוון משימות רחב, לרבות המרה בין סוגי נתונים, עיבוד תאריך/שעה, ניתוח סטטיסטי של תכני טבלאות ושאילתות, פעולות מתמטיות וטיפול בטקסט. תוכל להרחיב את ידיעותיך אודות פונקציות אם תלמד להכיר מה ניתן לעשות באמצעותן. Microsoft יצרה פונקציות רבות, ולכן אין צורך לכתוב אותן מחדש. לאחר מכן תוכל לעמוד על הדרך שהפונקציות משלימות זו את זו. באפשרותך גם ללמוד על תכונות הערכים המוחזרים. לדוגמה, הפונקציה Hex מחזירה מחרוזת שמייצגת את השקול ההקסדצימלי של מספר עשרוני, והפונקציה Oct מבצעת המרה דומה על מספרים אוקטליים. פלט שתי הפונקציות מתאים לשרשור ולהשוואת מחרוזות, אך לא לפעולות חשבוניות.

באפשרותך גם לחקור את השפעת ארגומנט אופציונלי של פונקציה על ערך ההחזרה של הפונקציה. אי-הבנה של ארגומנטים אלה עלולה לגרום לתוצאות מוטעות או מבלבלות. לדוגמה, הפונקציה strComp משווה שתי מחרוזות זו לזו. שני הארגומנטים הראשונים הם המחרוזות המיועדות להשוואה. הארגומנט השלישי הוא אופציונלי ומציין את סוג ההשוואה שיש לבצע. באפשרותך לציין השוואה תלוית-רישיות, השוואה שאינה תלויה ברישיות או לחילופין השוואה המבוססת על האפשרות **סדר מיון מסד נתונים חדש** (New Database Sort Order) בכרטיסיה **כללי** (General) של תיבת הדו-שיח **אפשרויות** (Options) של Access 2000. אם לא תציין את ההשוואה

האופציונלית, הפונקציה תנקוט את שיטת ההשוואה המצוינת במשפט Option Compare באזור ההצהרות (declaration) של המודול (אם לא צוינה שיטה כלשהי, Access ייבצע השוואה בינארית).

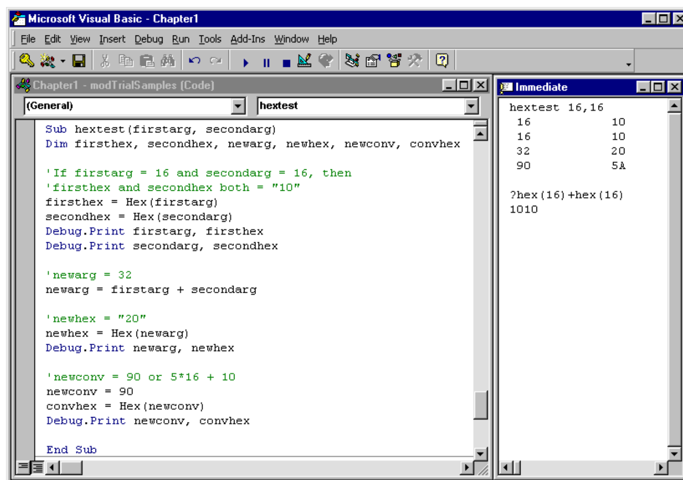
הטבלה שלפניך מציגה מבחר קטגוריות פונקציות עם תיאור ייעודה של הקטגוריה. הטבלה אינה כוללת את כל הקטגוריות, וכל קטגוריה מכילה בדרך-כלל יותר פונקציות. סיווג הפונקציות בקטגוריה כזו או אחרת הוא מעשה שרירותי, מכיון שפונקציות רבות מתאימות לכמה קטגוריות. לדוגמה, הפונקציה DateSerial המטפלת בתאריכים, מחזירה מספרים סידוריים של תאריך בהתבסס על ארגומנטים שאינם מסוג תאריך/שעה, אך באפשרותה גם להמיר תאריכי מחרוזת למספרים סידוריים. לפיכך הפונקציה יכולה להימנות בו-זמנית עם הקטגוריות תאריך/שעה והמרה.

קטגוריות נבחרות של פונקציות

קטגוריית פונקציה	פונקציות מייצגות	ייעוד
המרה	CDbl, CSng, CInt, CCur, CDec, Str, Val, Hex, Oct	קביעת סוג הפעולה החשבונית שביטוי מבצע והמרה בין סוגי נתונים
תאריך/ שעה	Date, Now, DateAdd, DatePart, DateDiff, Year, Month, Day, Hour, Minute, Second, Weekday, DateValue, DateSerial, MonthName, WeekdayName, FormatDateTime	ביטוי, חישוב, חילוץ ערכי תאריך ושעה מתוך מספרים סידוריים של תאריך וייצוגי מחרוזת של תאריכים ושעות.
צבירה המופעלת על תחום	DLookup, DCount, DSum, DAvg, DVar	חישוב ערכים סטטיסטיים, כגון ספירה, עבור תחום (לדוגמה, טבלה או שאילתה)
טיפול בשגיאות	Error, CVErr, IsError	לכידת שגיאות וקודי שגיאות מותאמים אישית
בדיקה	IsDate, IsTime, IsNumeric, IsNull, IsEmpty,VarType	הערכת סוגי נתונים באמצעות קוד
מתמטי	Rnd, Sqr, Exp, Log, Sin, Cos, Tan	ביצוע פעולות מתמטיות
הודעות	MsgBox, InputBox	הפקת הודעות ואיסוף נתוני קלט מהמקלדת
טקסט	Left, Right, Mid, Trim, UCase, LCaseStrReverse, Replace, InStr,InStrReverse	עיבוד מחרוזות
שונות	Sum, Count, RGB, FV, NPV, CreateObject, GetObject,SysCm	צבירת ערכים סטטיסטיים, קביעת מאפייני צבע, חישוב ערכים פיננסיים, יצירה או קליטת הפניות אל אובייקטי ActiveX, הצגת מדי התקדמות ועוד

פונקציות אלו מבצעות מיגוון משימות. פונקציות מחרוזת יכולות לחלץ, להמיר ולעגל חלקים לא רצויים של מחרוזת טקסט, אך באפשרותך גם לנצל את הפונקציות המוכללות לחישוב תוצאות פיננסיות או ערכים מתמטיים. פונקציות בדיקה שימושיות לקביעת סוגי נתונים לפני חישוב הביטויים. שימוש מושכל בפונקציות אלו ובפונקציות ההמרה יאפשר לך להימנע משגיאות אי-התאמה אפשריות בין סוגי נתונים (טיפול בשגיאות הוא חלק חיוני של פתרון כולל כלשהו. בלעדיו הפתרונות שתיצור לא יוכלו ללכוד שגיאות זמן ריצה, או לעורר שגיאות מותאמות אישית של זמן ריצה). פונקציות צבירה המופעלות על תחום (domain aggregate functions) אינן מצטיינות בביצועי מהירות מעולים, אך ביכולתן להחליף שאילתה שלמה או משפט SQL מורכב יותר. לדוגמה, יישומיך יכולים לנצל את פונקציות הצבירה לתחום, בשגרות VBA וגם בשאילתות ובטפסים. פונקציות צבירה ל- SQL, כגון Count ו-Sum יכולות אף הן להפיק תוצאות סטטיסטיות של מקורות רשומות. פונקציות אלו אינן ישימות בהיקף רחב כמו פונקציות צבירה לתחום. לדוגמה, לא ניתן להשתמש בהן בשגרות VBA.

תרשים 1.25 מציג שיגרה ואת החלון **Immediate** כדי להמחיש את התנהגות הפונקציה Hex. השיגרה הופכת את שני הארגומנטים למחרוזות הקסדצימליות ולאחר מכן מדפיסה את התוצאות בחלון **Immediate**.



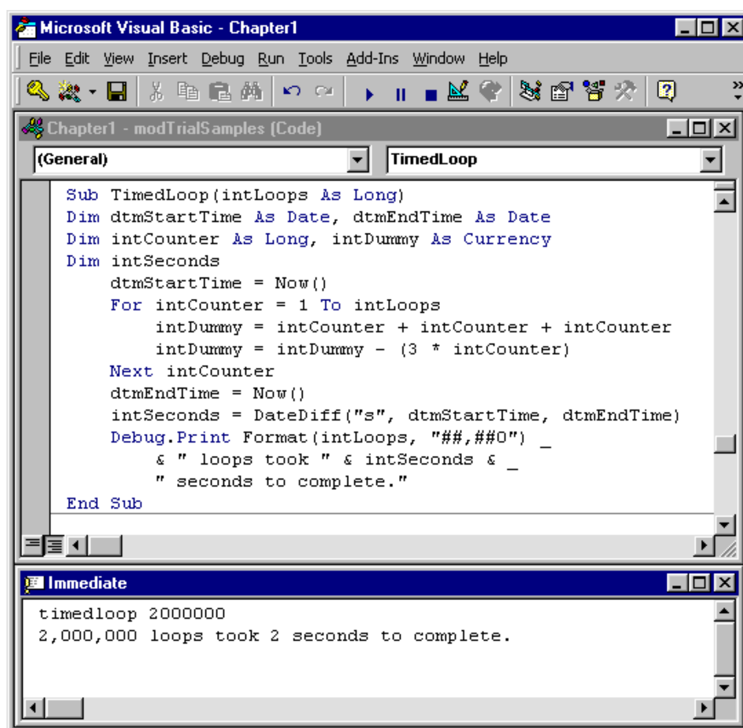
תרשים 1.25: שיגרה שבודקת את הפונקציה Hex

לא ניתן לבצע טיפול מתמטי במחרוזת המוחזרת על ידי הפונקציה Hex. הפונקציה hextext מתגברת על הבעיה על ידי ביצוע הפעולה החשבונית באמצעות הארגומנטים העשרוניים שלה, והמרת התוצאה שמתקבלת בעזרת הפונקציה Hex. שורות הקוד בהמשך השיגרה מדגימות גישה זו: הן מחברות את firstarg ואת secondarg זה לזה ושומרות את התוצאה ב-newarg. לאחר מכן, השיגרה הופכת את newarg למחרוזת הקסדצימלית ומאחסנת אותו ב-newhex. לבסוף, היא מדפיסה את newarg ואת newhex בחלון **Immediate**.

פעולת ההמרה האחרונה של hextext הופכת את הערך העשרוני 90 לערך ההקסדצימלי 5A. שורות ההערה הקודמות לפעולת ההמרה מציינות את העובדה ש-90 הוא תוצאת הביטוי 10 + 16 X 5.

זוג השורות האחרון בחלון **Immediate** מראה שהפונקציה Hex מחזירה ערכי מחרוזת. למעשה Hex מחזירה נתון מסוג variant, אך סוג המשנה שלו הוא מחרוזת. חיבור המספרים ההקסדצימליים 16 + 16 היה אמור לתת תוצאה של 20 הקסדצימלי, אך בחלון **Immediate** מקבלים 1010. תוצאה זו מעידה שסימן החיבור בשורה newarg = firstarg + secondarg ביצע שרשור מחרוזות במקום חיבור אריתמטי של מספרים.

עליך לחפש בהתמדה דרכים לצמצום זמן העיבוד. לדבר חשיבות מיוחדת בפתרונות שנועדו לשרת את משתמשיהם תקופה ארוכה, או כאלה שמבצעים משימות עסקיות חיוניות. תרשים 1.26 מציג שיגרה שיכולה לאמוד משך זמן ביצוע של קוד. השיגרה מחשבת את הזמן הדרוש לביצוע לולאה מספר קבוע של פעמים. באפשרותך להרחיב את התפקוד הבסיסי ולהוסיף לו השוואת זמנים של שני פתרונות תכנות. כפי שניתן לראות בחלון **Immediate**, קטע הקוד הפשוט פועל במהירות. מחשב הבדיקה שלי ביצע את הלולאה שני מיליון פעם בשתי דקות.



תרשים 1.26: הערכת זמן ביצוע של משימה. השיגרה TimeLoop מציגה, בין היתר, כיצד פועלות פונקציות מוכללות אחדות

השיגרה TimeLoop קולטת ארגומנט יחיד שקובע את מספר הפעמים שיש לבצע את הלולאה. ביצוע חוזר של לולאה מספר פעמים ידוע, מאפשר לעיתים קרובות להבחין בהבדלי ביצועים בין מחשבים. הלולאה הפשוטה שלנו כוללת שני משפטים קצרים בלבד. שים לב שלפני שהשיגרה מתחילה בביצוע הלולאה, היא שומרת את השעה הנוכחית (Now) ב-dtmStartTime. לאחר שהסתיימו כל החזרות, השיגרה שומרת את זמן הסיום ב-dtmStartTime. הפונקציה DateDiff מחשבת את ההפרש בין שני הערכים בשניות (ראינו קודם כיצד פונקציה זו מחשבת את ההפרש בין ימים ביחידות של ימים. הפונקציה מטפלת ביחידות זמן שונות, כולל שנים, רבעונים, שעות, שבועות ואפילו סופי שבוע). הפונקציה DateDiff מטפלת בהמרה לשניות ובכל התופעות המוזרות המתעוררות כתוצאה מהמעבר של חצות הלילה, ומחלצת את הערכים הרלוונטיים של מועדי ההתחלה והסיום.

בדיקה זו של משך זמן מתבססת על מספר גדול מאוד של חזרות, ולכן נוח במיוחד לבטא את מספרן באמצעות פסיקים. הפונקציה Format מאפשרת לבצע משימה זו.

פונקציות צבירת התחום מהוות קבוצה ייחודית. הן מחזירות נתונים סטטיסטיים ונתוני חיפוש על תחום (קבוצת רשומות). הרשומות יכולות להימצא בטבלה אחרת, או להיות תוצאה של שאילתת בחירה. הפונקציות יכולות להחזיר נתונים שונים על תחום נתון, כגון: ספירה, סכום, ממוצע, שונות וסטיית תקן. תוכל לנצל את פונקציות צבירת התחום DMin ו-DMax כדי להחזיר את הערך הקטן או הגדול ביותר בעמודה שבתחום. פונקציות צבירת התחום DFirst ו-DLast מחזירות רשומה אקראית במקום רשומה ראשונה או אחרונה מתוך רשומות הממוינות לפי שדה מסוים. פונקציית צבירת התחום DLookup היא פונקציה מקובלת מאוד; היא מחזירה רשומה אחת או יותר שתואמת לקריטריון נתון. לכל פונקציות הצבירה לתחום ישנה תבנית הדומה לתבנית הבאה:

```
DFunctionName("fieldexpression", "domainname", "criteria")
```

כל השמות של פונקציות צבירה המופעלות על תחום מתחילים ב-"D". שים לב שכל שלושת הארגומנטים של הפונקציה מופיעים בין מרכאות כפולות. ניתן, אך לא חובה, להציב את שם השדה של הארגומנט הראשון והשלישי בין סוגריים. שני הארגומנטים הראשונים הם חובה, והשלישי – אופציונלי. הארגומנט fieldexpression מציין שדה שעליו יש לדווח. הארגומנט domainname הוא שם מקור הרשומות. הארגומנט האופציונלי השלישי מציין משפט קריטריון כדי לציין את האלמנטים שברצונך לכלול בסדרת הרשומות המוחזרת (קבוצת ההחזרה). כדי להחזיר את שם החברה מהרשומה הראשונה בטבלה Shippers שבמסד הנתונים Northwind, הפעל את הפונקציה DLookup הבאה:

```
DLookup("CompanyName", "Shippers", "ShipperID = 1")
```

הקריטריון בדוגמה זו מציין את הרשומה בטבלה שבה שדה ShipperID מכיל את הערך 1. זהו שדה מספור אוטומטי המכיל נתונים מסוג Long Integer. אם קריטריון מסוג זה מתאים לצרכיך, הדבר יפשט את השימוש בפונקציות צבירה לתחום. גירסה

זו מחזירה את ערך השדה מתוך הרשומה היחידה התואמת לקריטריון. אם קיימות כמה רשומות התואמות לקריטריון, הפונקציה תחזיר את הרשומה התואמת הראשונה.

פונקציות צבירה נוספות לתחום, כגון DSum, DCount ו-DAvg מחשבות תוצאות המבוססות על רשומה אחת או יותר. משפט VBA שלפניך מדפיס את תוצאת ספירת החשבוניות שמועד פירעונן חלף בחלון **Immediate** בהתאם לתוצאת החישוב של הפונקציה DCount:

```
Debug.Print DCount("InvoiceDate", "InvoiceDates", "InvoiceDate < Now()-30")
```

דוגמה זו מחשבת את מספר הרשומות שמועד פירעונן חלף ומדפיסה את תוצאת החישוב. כשמפעילים אותה על הטבלה InvoiceDates שבקובץ הדוגמה בפרק 1, בתאריך 12/9/98, הפונקציה DCount מחזירה שתי רשומות. ייתכן שתרצה להוסיף רשומות חדשות לטבלה, או לשנות תאריכים כלשהם כדי לקבל תוצאה אחרת של ספירת הרשומות עבור דוגמה זו.

תרשים 1.27 מציג טופס שמאפשר למשתמש לקבוע את ערכי הגוונים אדום, ירוק וכחול של צבע ולאחר מכן להציג את הצבע המתקבל כשהמשתמש לוחץ על תיבת הטקסט שבצד ימין. גווני האדום, הירוק והכחול של הצבע יכולים לקבל ערכים בטווח בין 0 ל-255. אם כל שלושת הערכים הם 0, יתקבל צבע שחור; אם כולם 255, יתקבל צבע לבן. קביעת אחד הגוונים בתור 255 בעוד ששני הגוונים האחרים 0, נותנת צבע חזק של הגוון שקיבל את הערך המקסימלי. הגדרות הגוון במחשב שלי מציגות את הצבע המוגדר בתרשים 1.27 בתור אדום בהיר או ורוד.

frmColorCalculator : Form
Color Calculator
Ener a number from 0 through 255 into each box below, then...
click the box below to see your color!
Red: 255
Green: 150
Blue: 150

תרשים 1.27: טופס המראה את מציג הצבע

שגרת האירוע שלפניך השייכת לתיבת הטקסט הימנית בתרשים 1.27, מבצעת את תהליך הטיפול הפשוט במציג הצבע. מלבד בדיקת ערכים ריקים בתיבות הטקסט של גווני הצבע, השיגרה מכילה משפט בודד. המשפט מנצל את הפונקציה RGB כדי להגדיר את המאפיין BackColor (צבע רקע) של תיבת הטקסט הגדולה בחלקו הימני של הטופס.

```
Private Sub txtColorMe_Click()
'If null color field, set a default value.

    If IsNull(Me.txtRed.Value) Then
        Me.txtRed.Value = 0
    End If

    If IsNull(Me.txtGreen.Value) Then
        Me.txtGreen.Value = 0
    End If

    If IsNull(Me.txtBlue.Value) Then
        Me.txtBlue.Value = 0
    End If

'Set BackColor property.
    Me.txtColorMe.BackColor = RGB(Me.txtRed.Value, Me.txtGreen.Value, _
        Me.txtBlue.Value)
End Sub
```

בדיקת ערכים ריקים (Null) חיונית, כיון שערכים כאלה יגרמו לפונקציה RGB להיכשל. ניתן לתקוף את הבעיה בדרכים רבות, וקוד השיגרה txtColorMe_Click הוא רק אחת מהן. לדוגמה, קביעת ערך ברירת המחדל בתור 0 במקרה שלא צוין ערך כלשהו, היא קביעה שרירותית בעליל; הערך יכול היה להיות 255 או מספר כלשהו בין 0 ל-255. אפשרות נוספת היא להימנע מחישוב הפונקציה RGB, ובמקום זאת לבקש מהמשתמש למלא ערכים בכל שלוש תיבות הטקסט. והדרך האחרונה – לאפשר התרחשות של שגיאת זמן ריצה עקב ערך Null, ללכוד את השגיאה ואז לטפל בה בהתאם. בסעיף הבא נציג דרכים למימוש גישה זו.

ניפוי באגים ולכידת שגיאות

אין קוד נקי משגיאות. מלבד שגיאות תחביר ושגיאות לוגיות, נחלת כל פתרון תוכנה בשלבי הפיתוח, גם קוד שאמור לתפקד כהלכה "מניב" לעיתים שגיאות זמן ריצה. שגיאות אלו עשויות להיות תוצאה של אי-הקפדת המשתמש להזין ערך, או הזנת ערך מחוץ לתחום. הפעלות נסיוניות מאפשרות לפתור את רוב הבעיות מסוג זה, אך אינך יכול למנוע מהמשתמש להזין ערך בלתי צפוי כלשהו, או ללחוץ על לחצן בטרם עת.

לכידת שגיאות הוא תהליך "תפיסת" השגיאות המתרחשות בזמן ריצה, בטרם תגרומנה לסיום לא מבוקר של היישום. לכידת שגיאות מעניקה לך שליטה בעניינים. הקוד יכול לבדוק את השגיאה ולעיתים אפילו להתגבר על הבעיה מבלי שהמשתמש יצטרך לעשות כלום. באפשרותך לקבוע את נחיצות התערבות המשתמש ולהציג לו הודעה בהתאם, ואף לקבל ממנו קלט חדש באמצעות תיבות דו-שיח מותאמות אישית ולאחר מכן להמשיך את פעולת היישום. גם במצבים בהם לא ניתן להתאושש בצורה מסודרת ממצב שגיאה, ניתן עדיין לסיים את התוכנית בתנאים שנקבעו מראש.

הפתרון יכול לשמור מידע על משפט הקוד בו התרחש הכשל, בצירוף קוד השגיאה, תיאור וכל פרטי המידע הרלוונטיים. אחר כך תוכל להורות למשתמש לדווח על התקלה לגורם המתאים.

מאחר ש-Access מאפשר למפתחים להגיב לשגיאות זמן ריצה, מומלץ שתחשוב על לכידת מצבי השגיאה ביישום שלך, כגון הזנת ערכים לא חוקיים. לאמיתו של דבר, Access מאפשר ליצור שגיאות מותאמות אישית בדרכים שונות. הדבר מאפשר לקוד להתאושש מהשגיאות המותאמות אישית שיצרת באותו אופן שהוא נוהג בשגיאות רגילות של Access. קרוב לוודאי שתדע טוב יותר כיצד להגיב לשגיאות המותאמות אישית, מכיון שאתה יצרת אותן.

לכידת שגיאות – תחביר בסיסי

כדי לממש מנגנון ללכידת שגיאות, עליך להבין את המשפט On Error, את האובייקט Err ואת המשפט Resume. כל אלה פועלים יחד כדי לאפשר לכידת שגיאות ולסייע לך לטפל בשגיאות לאחר לכידתן.

קיימות שלוש גרסאות למשפט On Error, אך הגירסה שלפניך מאפשרת לכידת שגיאות וקרוב לוודאי שתבחר להשתמש בה:

```
On Error GoTo linelabel
```

כש-Access מזהה שגיאה, משפט זה מעביר את השליטה במצב לתוכנית שלך, במקום לתת ל-Access לטפל בשגיאה לפי ברירת המחדל שלו. הארגומנט linelabel הוא שם השורה שאליה ברצונך להעביר את השליטה. במקום ש-Access ייבצע את תהליך הטיפול המוגדר מראש, יתבצע הקוד המתחיל בשורה ששמה צוין במשפט. על הקוד לקבוע ולפענח את מספר השגיאה, כדי שהתוכנית תנקוט פעולה הולמת. אם אין באפשרותך לתקן את התקלה, תוכל לפחות לקבל מידע כלשהו אודותיה בטרם תסתיים התוכנית בצורה מסודרת.

האובייקט Err מלמד על השגיאה. לאובייקט שני מאפיינים. המאפיין Number מגדיר את השגיאה במדויק. המספרים המאוחסנים במאפיין Number אינם משתנים מגירסה לגירסה של Access. משמעות הדבר היא שניתן לנצל את המאפיין בתור מזהה שגיאות עמיד. מידע נוסף ניתן להפיק מהמאפיין Description של האובייקט Err שמכיל תיאור מילולי של השגיאה. עליך להציג את תיאורי השגיאות בחלון המיידי או בתיבת טקסט.

כאשר מתכננים תוכנית מראשיתה, תיאורי האובייקט Err יכולים לסייע לשפר את התכנון ולפתח תגובות הולמות לשגיאות זמן ריצה שגרתיות, כגון ערך קלט מחוץ לתחום משתנה. גירסה מוכנה לשימוש של התוכנית יכולה גם לשמור תיאורי שגיאה בקובץ יומן. הדבר מאפשר לתכנן תיקוני קוד מהירים לאורך חיי היישום. יחד עם זאת, בדרך-כלל לא תרצה לזהות שגיאות על פי תיאוריהן שהם ארוכים ממספריהן (בנוסף, Microsoft עשויה לנסח מחדש תיאורי שגיאה, למרות שמספריהם אינם משתנים).

המשפט Resume משמש לציון נקודת המשך הפעלת היישום לאחר זיהוי וטיפול בשגיאה. אם ברצונך להעביר את השליטה למשפט שגרם לשגיאה, השתמש במילה Resume ללא ארגומנטים כלשהם. גישה זו עשויה להתאים למקרים בהם אתה מתקן את התקלה שגרמה לשגיאה מלכתחילה, כגון ערך שדה חסר. אם מנגנון הטיפול בשגיאות אינו מסוגל להתגבר על השגיאה, ייתכן שתרצה למרות זאת להמשיך בביצוע התוכנית שורה אחת אחרי זו שגרמה לשגיאה. במצב זה השתמש במשפט Resume Next. כל עוד השגיאה אינה משפיעה על תקפות שורות הקוד הנותרות, גישה זו באה בחשבון. במצבים שהיישום מגלה שגיאה שאינו מסוגל לטפל בה, ייתכן שתרצה פשוט להפסיק את פעולתו באופן מיידי. במקרים אלה העבר את השליטה למשפט Resume linelabel. בעת לכידת שגיאות, תכלול לעיתים קרובות משפט Exit Function או משפט Exit Sub בצירוף תווית. נוח להשתמש בתווית זו בתור ארגומנט המשפט Resume linelabel במקרים שברצונך לצאת מהשיגרה לאחר שגיאה.

דוגמאות ללכידת שגיאות

סעיף זה מציג שתי דוגמאות של לכידת שגיאות. הראשונה שביניהן היא יישום בסיסי של לוגיקת לכידת שגיאות. הדוגמה מציגה לכידת שגיאות של שגרת פונקציה בתיקיה Module. היא מציגה גם כיצד לעורר קודי שגיאות מותאמים אישית. הדוגמה הראשונה ממחישה את קלות הטיפול בשגיאות היישום הפנימיות באמצעות לוגיקה זהה לזו המשמשת ללכידת שגיאות ב-Access. הדוגמה השנייה מנצלת לוגיקת לכידת שגיאות בקוד התומך בטופס. התכונה הבולטת של יישום זה היא תגובתו השונה לכל אחת משתי מחלקות של שגיאות.

שגרת הפונקציה שלפניך משלבת לכידת שגיאה בדוגמה הפשוטה יותר שהוצגה קודם לכן בתרשים 1.16. שים לב כי שיגרה זו מבצעת אחת מארבע פעולות חשבוניות. מעבירים לשיגרה שלושה ארגומנטים: שני אופרנדים ומחרוזת המציינת את הפעולה החשבונית שיש לבצע על המספרים שהועברו אל הפונקציה.

שתי שגיאות לפחות עלולות להתעורר מפעולת הקוד בזמן ריצה. הראשונה, המשתמש עלול לנסות לחלק מספר ב-0. פעולה זו יוצרת שגיאה מספר 11. שגיאה שנייה, הפעולה עלולה ליצור תוצאה גדולה או קטנה מזו שמאפשר סוג הנתונים. סוג שגיאה זה נקרא במונחי Access **שגיאת גלישה** (overflow error) ומספרה הוא 6. מנגנון לכידת השגיאות של השיגרה שלפניך לוכד את שתי השגיאות הללו במפורש, ובאופן משתמע מניח מלכודת לכל סוג אחר של שגיאה. קטע הטיפול בשגיאה מסייע למשתמש, על ידי הצעת תיקון לחילוק ב-0, וגם מסביר מה טיבה של שגיאת גלישה בשפה ידידותית יותר מזו ש-Access משתמש. הואיל ולעיתים קרובות תכתוב יישומים עבור קהל משתמשים מוכר לך, תוכל לפנות אליהם באופן ישיר יותר מאשר תוכנית כללית (כמו Access).

```
Public Function Computer2(dblNumber1, dblNumber2, Operation As String)
On Error GoTo Computer2Handler

   Select Case Operation
      Case "Addition"
         Computer2 = dblNumber1 + dblNumber2
      Case "Subtraction"
         Computer2 = dblNumber1 - dblNumber2
      Case "Multiplication"
         Computer2 = dblNumber1 * dblNumber2
      Case "Division"
         Computer2 = dblNumber1 / dblNumber2
   End Select

Computer2Exit:
   Exit Function

Computer2Handler:
   If Err.Number = 11 Then
      MsgBox "Can't divide by zero. Change second number from 0.", _
      vbInformation, "Programming Microsoft Access 2000"
   ElseIf Err.Number = 6 Then
      MsgBox "Result exceeds data type value limits.", vbInformation, _
         "Programming Microsoft Access 2000"
   Else
      MsgBox Err.Number & ": " & Err.Description, vbInformation, _
         "Programming Microsoft Access 2000"
   End If
   Resume Computer2Exit
End Function
```

המשפט On Error מופיע מייד לאחר תחילת התוכנית. הוא ממוקם שם כדי ללכוד שגיאות מוקדם ככל האפשר. אם פונקציה אחרת קוראת לתוכנית הנוכחית, ייתכן שכדאי להשתמש בלכידת שגיאות באותה תוכנית, כך שהיא תוכל להשתמש בלכידת שגיאות עבור הארגומנטים שלה המועברים לפונקציה Computer2.

אם תקליד ?Computer2(2, 4, "multiplication") בחלון **Immediate** ותקיש Enter, הפונקציה תחזיר את הערך 8. החלף את פעולת הכפל בפעול חילוק (division), והערך החוזר יהיה 0.5. כעת החלף את הארגומנט השני ב-0. לולא שילבת מנגנון לכידת שגיאות, התוכנית היתה "נופלת" בצירוף הודעת מערכת מתאימה. מנגנון הטיפול בשגיאות מעביר את השליטה ל- Computer2Handler ברגע שנעשה ניסיון לחלק ב-0. הרוטינה בודקת את שגיאה מספר 11, ומכיון שזו השגיאה שאירעה, היא מציגה למשתמש הודעה שעליו לשנות את גורם החילוק למספר שונה מ-0. הקלד

?Computer2(1.79E308, 4, "multiplication") והקש Enter כדי לגרום לשגיאת גלישה. פעולת הכפל מנסה ליצור תוצאה גדולה מזו ש-Access מסוגל לייצג. גם במקרה זה מועברת השליטה ל- Computer2Handler מייד לאחר שהפעולה החשבונית נכשלת. לאחר שנקבע כי קוד השגיאה אינו 11, לוגיקת לכידת השגיאות בודקת את קוד השגיאה כנגד 6. התאמת הקוד גורמת להצגת תיבת הודעה שמסבירה את סיבת השגיאה.

קטע הטיפול בשגיאות של השיגרה משתמש במשפט If...Then...ElseIf...Else. הפסוקית Else מציעה דרך ללכידת שגיאות שהקוד אינו לוכד באופן מפורש. בעת אתחול היישום, אינך מסוגל לדעת בדיוק אילו שגיאות תתרחשנה. במקרה זה השתמש במשפט If...Then...Else ללא פסוקית ElseIf כלשהי. הפסוקית Else מציגה מספרי שגיאות בצירוף תיאוריהן כשהן מתרחשות בשלב הבדיקה. באפשרותך לנצל מידע זה כדי לפתח מנגנוני לכידה מפורשים עבור שגיאות מסוגים מסוימים יחד עם פתרונות זמינים למצבים אלה. במהלך בדיקת היישום ייתכן שתהיינה לך הזדמנויות נוספות לשכלל את אוסף הפסוקיות ElseIf של מנגנון הטיפול בשגיאות.

השורה האחרונה בקטע הטיפול בשגיאות היא המשפט Resume. משפט זה מעביר את השליטה לשורה שממנה יוצאים מהשיגרה. באפשרותך גם להשתמש במשפטי Resume נפרדים לכל אחת ממלכודות השגיאות. גישה זו מתאימה למקרים בהם היישום מחייב אותך לנקוט פעולות שונות לכל סוג של מלכודת שגיאות.

מנגנון הלכידה לא ילכוד סוגי שגיאות אחדים. לדוגמה, הוא לא ילכוד שגיאה שמקורה באיות שגוי של הפעולה החשבונית, כגון multiplcation במקום multiplication. שגיאת מפעיל מעין זו עוברת דרך משפט Select Case מבלי לעצור כדי לבצע פעולה חשבונית כלשהי, עד שהיא מגיעה למשפט Exit Function. יש צורך ללכוד שגיאה זו ולהודיע למשתמש מה עליו לעשות. הפתרון הוא לזהות את הבעיה בעזרת הפסוקית Case Else במסגרת המשפט Select Case. לאחר מכן מעוררים שגיאה מותאמת אישית שמודיעה למשתמש כיצד לנהוג. לוגיקת הלכידה עמידה מאוד, ולכן עליך לשנות רק את המשפט Select Case. הקטע הבא מתוך הקוד המתוקן מציג גירסה חדשה, הכוללת רק שני משפטים נוספים:

```
Select Case Operation
    Case "Addition"
        Computer2 = dblNumber1 + dblNumber2
    Case "Subtraction"
        Computer2 = dblNumber1 - dblNumber2
    Case "Multiplication"
        Computer2 = dblNumber1 * dblNumber2
    Case "Division"
        Computer2 = dblNumber1 / dblNumber2
    Case Else
        Err.Raise 1, , "Wrong operation."
End Select
```

השורה שלאחר Case Else מציגה את התחביר לייזום קוד שגיאה מותאמת אישית. שגיאה מספר 1 אינה מקבלת הצבה מ-DAO, VBA או ADO. לכן, כאשר המשפט Case Else לוכד פעולה ששמה מאוית בצורה שגויה, הוא מעורר שגיאה מותאמת אישית וזו מעבירה את השליטה למשפט Computer2Handler. הפסוקית Else בקטע משפטים זה מזהה את קוד השגיאה 1 ויוצרת תיבת הודעה ובה קוד השגיאה והתיאור המותאם אישית "Wrong operation" (פעולה בלתי חוקית). לאחר שנסקור את תהליך לכידת שגיאות בעזרת טופס, נחזור לנושא בחירת השגיאה המותאמת אישית.

הקוד התומך בטופס שמוצג בתרשימים 1.18 עד 1.20 מתבסס על משפטי For Each...Next מקוננים כדי לטפל בתגובות לטופס. כזכור, שגרת אירוע הלחיצה עבור לחצן הפקודה עוברת בלולאה על כל הפקדים בטופס כדי למצוא את שתי תיבות הטקסט. הדבר נעשה כדי להימנע משגיאה המתרחשת במקרה שהפונקציה IsNull מופעלת על פקד, כגון תווית או לחצן פקודה, שאינו תומך בפונקציה זו.

גישה חלופית היא לאפשר לשגיאה להתרחש, ללכוד אותה ולהתאושש מהבעיה. הפעלת הפונקציה IsNull על פקד שאינו מתאים לה גורמת לשגיאה מספר 438. לפניך הגירסה החדשה של שגרת האירוע cmdSubmit_Click.

```
Private Sub cmdSubmit_Click()
On Error GoTo SubmitErrorTrap
' Check for valid entries.
    For Each ctl In Screen.ActiveForm.Controls
        If IsNull(ctl.Value) Then
            MsgBox "Please enter information in both input boxes.", _
                vbInformation, "Programming Microsoft Access 2000"
            MarkFieldsToEdit
            Exit For
        End If
SubmitNextCtl:
    Next ctl

SubmitExit:
    Exit Sub

SubmitErrorTrap:
    If Err.Number = 438 Then
        Resume SubmitNextCtl
    Else
        MsgBox Err.Number & ": " & Err.Description, vbInformation, _
            " Programming Microsoft Access 2000"
        Resume SubmitExit
    End If
End Sub
```

שים לב כי הלוגיקה של לכידת השגיאות מאריכה את הקוד, למרות שהיא מאפשרת להסיר אחד משני משפטי For Each...Next. הלוגיקה החדשה ללכידת שגיאות עמידה יותר. לולאות For המקוננות נמנעות מסוג שגיאה אחד בלבד – שגיאה מספר 438. להלכה, החלופה החדשה יכולה לטפל בכל סוג שגיאה. בנוסף, היא מסוגלת להגיב לשגיאות בשתי דרכים שונות. שים לב כי שגיאה 438 מגיבה בהעברת השליטה אל המשפט Next בחלק שנותר בלולאת For. הדבר מאפשר לתוכנית להמשיך לטפל בפקדים נוספים שתואמים לפונקציה IsNull. שגיאה מסוג אחר כלשהו גורמת לשגרת האירוע להסתיים לאחר הצגת תיבת הודעה ובה מספר השגיאה הבלתי צפויה ותיאורה.

העלאת שגיאות

הפונקציה Raise (העלאה) של האובייקט Err עמידה, אך היא מחייבת לאפשר את הטיפול בשגיאות. הוספת רוטינות טיפול בשגיאות עלולה להאריך את הקוד. אם אין לך ניסיון רב בכתיבת קוד וברשותך שיגרה קצרה שיש לה פוטנציאל קטן לגרום לשגיאות, ייתכן שתרצה דרך פשוטה יותר להחזיר קוד שגיאה.

השיגרה שלפניך מבצעת זאת בעזרת הפונקציה CVErr. שיטה זו אינה עמידה לתקלות כמו מנגנון שלם לטיפול בשגיאות, מכיון שהיא לוכדת שגיאות מסוג אחד בלבד. הואיל ולא קיים מנגנון לטיפול בשגיאות, היישום כולו עלול לקרוס אם מתרחשת שגיאת זמן ריצה, אך קל ופשוט להשתמש בפונקציה CVErr. הפונקציה אינה מתנגשת עם קודי שגיאה מוכללים, מכיון שאינה מחזירה קודי שגיאה באותו מסלול. עליך לשקול את יתרונות וחסרונות השימוש בפונקציה CVErr במקרים המחייבים החזרת קוד שגיאה.

```
Public Function Computer1(dblNumber1, dblNumber2, Operation As String)
    Select Case Operation
        Case "Addition"
            Computer1 = dblNumber1 + dblNumber2
        Case "Subtraction"
            Computer1 = dblNumber1 - dblNumber2
        Case "Multiplication"
            Computer1 = dblNumber1 * dblNumber2
        Case "Division"
            Computer1 = dblNumber1 / dblNumber2
        Case Else
            Computer1 = CVErr(2002)
    End Select
End Function
```

הפונקציה לוכדת שגיאה של פעולה שאויתה בצורה שגויה או כזו שאינה נתמכת בקוד. המשפט Select Case מבודד בעיות אלו באמצעות הפסוקית Case Else. כשהמשתמש מקליד פעולה שגויה, הפונקציה מחזירה נתון מסוג Variant שסוג המשנה שלו Error

מכיל קוד שגיאה מספר 2002. גירסה זו אלגנטית יותר מהשיגרה Computer2 שהוצגה קודם לכן בפרק. יחד עם זאת, Computer2 לוכדת במפורש שגיאות מסוגי חלוקה באפס וגלישה, גם כשהיא מחזירה שגיאות מסוגים אחרים. השיגרה Computer1 קצרה בהרבה ממנה ולוכדת את הבעיה היחידה של פעולה שאויתה בצורה שגויה, או כזו שאינה נתמכת בקוד.

הפונקציה CVErr אינה מושפעת מהתנגשויות בין מספרי שגיאה של VBA ו-Access, אך זוג השגרות שלפניך מפשט בצורה משמעותית את תהליך איתור קודי שגיאה פנויים שתוכל לנצל לשימוש המותאם אישית שלך. הפונקציה הראשונה, VBADAOUsedErrorList, מציגה את קודי השגיאה בטווח המספרים שבשימוש VBA ו-DAO. עליך לציין את המספר הפותח והמספר המסיים בעת הקריאה לשיגרה. המספרים הנעדרים מהפלט זמינים לשימוש בתור קודי שגיאה מותאמים אישית. השיגרה השנייה מאפשרת אף היא לציין מספר פותח ומספר מסיים של הטווח שבו ניתן לאתר קודי שגיאה. שיגרה זו, לעומת זאת, מדפיסה בחלון המיידי את קודי השגיאה שאינם שמורים על ידי VBA או DAO. חלק גדול מקודי השגיאה של ADO הם מספרים שליליים גדולים.

```
Sub VBADAOUsedErrorList(intStart, intEnd)
Dim intErrorCode As Long, strAccessError As String

    For intErrorCode = intStart To intEnd
        strAccessError = AccessError(intErrorCode)
        If strAccessError <> "" Then
            If strAccessError <> _
                "Application-defined or object-defined error" Then
                Debug.Print intErrorCode, strAccessError
            End If
        End If
    Next intErrorCode

End Sub

Sub VBADAOUnUsedErrorList(intStart, intEnd)
Dim intErrorCode As Long, strAccessError As String

    For intErrorCode = intStart To intEnd
        strAccessError = AccessError(intErrorCode)
        If strAccessError = "" Or strAccessError = _
            "Application-defined or object-defined error" Then
            Debug.Print intErrorCode, strAccessError
        End If
    Next intErrorCode

End Sub
```

פקודות מאקרו

פקודות מאקרו אינן חלק מ-VBA, אך הן מאפשרות להפוך יישומי Access לאוטומטיים בצורה קלה ופשוטה. אחד היתרונות הגדולים של פקודות מאקרו הוא שאין צורך לזכור את התחביר שלהן; Access מאפשר לבחור בקלות פעולת מאקרו מתוך תיבת רשימה נפתחת. לעומת זאת, טכניקות ניפוי הבאגים של מאקרו אינן עשירות כמו אלו של VBA. חיסרון נוסף – פקודות מאקרו מאוחסנות במכולת מאקרו נפרדת, ולא מאחורי טפסים, מה שעלול לגרום במשך הזמן לבעיות תחזוקה. בנוסף, פקודות מאקרו של Access שונות מאלו של Excel, Word ו-PowerPoint. לאמיתו של דבר, פקודות מאקרו הולכות והופכות למיושנות.

אם השתמשת עד כה בגירסה קודמת של Access, יש לך בוודאי ניסיון מעשי בעבודה עם פקודות מאקרו. במהדורות Access קודמות, פקודות מאקרו היו האמצעי היחיד ליצירת אפשרויות אתחול מיוחדות ותפריטים מותאמים אישית. מעתה תוכל לנהל תפריטים באמצעות אובייקט האוסף CommandBars (סרגלי פקודות) של VBA. בנוסף, באפשרותך לנצל את תיבת הדו-שיח **הפעלה** (Startup) לשליטה במיגוון ההיבטים של אתחול יישום Access, כגון הצגת טופס בסיום תהליך אתחול, ציון אם המשתמש יכול לערוך שינויים בתפריטים ובסרגלי כלים, וקביעה אם יוצג חלון מסד הנתונים.

תכנון פקודות מאקרו

כדי לעבוד עם פקודות מאקרו, עליך לדעת כיצד להשתמש בממשק המאקרו וגם להכיר בצורה בסיסית את פעולות המאקרו. תרשים 1.28 מציג את חלון המאקרו לאחר טעינת מסד הנתונים Northwind.

כל מאקרו של Access מורכבת משורה אחת או יותר בחלון **מאקרו** (Macro). כל שורה מציינת פעולה (כגון העברת מיקוד לפקד), תנאי אופציונלי עבור הפעולה והערה אופציונלית. האזור **ארגומנטים של הפעולה** (Action Arguments) בתחתית החלון (מוצג בעת בחירת פעולה) מאפשר לציין את האפשרויות עבור כל פעולה ומציג עזרה עבור הפריט שנבחר. בנוסף, השורה הראשונה בכל מאקרו מציינת את שם המאקרו (בחר **שמות מאקרו** (Macro Names) מהתפריט **תצוגה** (View) של Access אם אינך רואה שמות מאקרו).

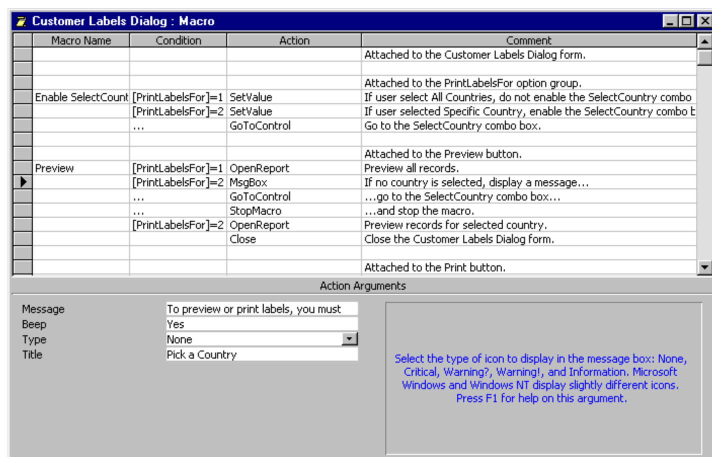
פעולות מאקרו רבות כוללות אפשרויות שניתן לבחור בהן באמצעות לחיצה פשוטה. לדוגמה, השורה השנייה של המאקרו Preview בתרשים 1.28 (השורה בעלת המשולש שקודקודו פונה ימינה) מציגה תיבת הודעה. באזור **ארגומנטים של הפעולה** בחלון ניתן לבחור בארגומנטים של תיבת ההודעה מתוך תיבות רשימה נפתחות. לדוגמה, תוכל להשתמש בתיבת הרשימה הנפתחת **סוג** (Type) כדי לבחור בסמל שיוצג בתיבת ההודעה.

בגרסאות קודמות, הבחירה בעזרת הצבעה ולחיצה העניקה לפקודות המאקרו יתרון מוחלט על קוד VBA מבחינת נוחות השימוש. כיום, VBA כולל את הרכיב IntelliSense.

אומנם קל להשתמש בפקודות מאקרו עקב קלות הבחירה של ארגומנטים, אך הן אינן מציעות את הממשק הגמיש שמאפשר להציג בו-זמנית את הקוד (לרבות הארגומנטים) של פקודת מאקרו בעלת שורות רבות.

העמודה החשובה ביותר בחלון **מאקרו** היא העמודה **פעולה** (Action). אינך צריך להקליד פעולות בעמודה זו; באפשרותך ללחוץ על שורה כלשהי בעמודה כדי לקבל תפריט מלא של פעולות מאקרו. בעמודה **תנאי** (Condition), משמאל לעמודה **פעולה**, תוכל להגדיר קריטריונים ששולטים באופן מותנה בביצוע הפעולה באותה שורה.

תוכל לאחסן פקודות מאקרו רבות באובייקט מאקרו יחיד. זו דרך אפקטיבית לאחסון פקודות מאקרו רבות המשתפות ביניהן תפקוד כללי. השתמש בעמודה **שם המאקרו** (Macro Name) כדי להקצות שמות לפקודות המאקרו שנמצאות באובייקט המאקרו. בעת הקצאת פקודת מאקרו לפעולה על פקד בטופס או בדוח, מציינים את המאקרו באמצעות שם המורכב משני חלקים: החלק הראשון הוא שם אובייקט המאקרו, והחלק השני הוא שם פקודת המאקרו המצוין בעמודה **שם המאקרו** (הפרד את שני חלקי השם בתו נקודה, כמו בשם MyMacros.DisplayHelp).



תרשים 1.28: המרכיבים העיקריים של ממשק המשתמש בחלון **מאקרו** (Macro)

Access כולל שגרת המרה מוכללת לתרגום פקודות מאקרו לקוד VBA. הפעלת רכיבי ההמרה משתנה במקצת אם היישום קורא לפקודות המאקרו מתוך טופס (או דוח), או מתוך חלון מסד הנתונים. במקרה הראשון, בחר באפשרות **מאקרו** (Macro) מתוך תפריט **כלים** (Tools) ולאחר מכן בחר באפשרות **המרת פקודות מאקרו של טופס ל-Visual Basic** (Convert Form's Macros To Visual Basic) או באפשרות **המרת פקודות מאקרו של דוח ל-Visual Basic** (Convert Report's Macros To Visual Basic). כדי להמיר מאקרו מתוך חלון מסד הנתונים, בחר אותו כרטיסיה **מאקרו**. בחר בתפריט **קובץ** (File), בחר **שמירה בשם** (Save As) ובתיבת הדו-שיח **שמיה בשם**, בחר **מודול** (Module) מתיבת הרשימה הנפתחת **כסוג** (As).

VBA לעומת פקודות מאקרו

VBA הולך ומשתפר במידה רבה מאוד ממהדורה למהדורה, בעוד שפקודות מאקרו נותרו כמעט ללא שינוי. זוהי הסיבה העיקרית לעבור מפקודות מאקרו ל-VBA, אך קיימות סיבות נוספות:

- מנגנון לכידת השגיאות המוכלל ב- VBA ורכיבי ניפוי באגים גמישים נוספים מקטינים באופן דרמטי את עלות מחזור החיים ואת עלות הבעלות על יישומי Access, עקב הקטנת עלויות התחזוקה שלהם.
- VBA מציע רכיבי עיבוד מתקדמים רבים, כגון אפשרויות עיבוד בלולאה, תפקודיות אינטרנט מעולה וקישורי OLE DB למקורות נתונים חיצוניים.
- הטופס המותאם אישית ומחלקות הדוח של VBA מפשטים את השימוש החוזר בקוד.
- VBA מאפשר להציג בו-זמנית ארגומנטים של כל השורות - לא רק של שורה אחת.
- VBA מאפשר להפעיל פונקציות API של Windows.

2

מודלי גישה לנתונים

Microsoft Access 2000 תומך בשני מודלים לגישה לנתונים: המודל המקובל **DAO** (Data Access Objects - אובייקטי גישה לנתונים) ו-**ADO** (ActiveX Data Objects - אובייקטי נתונים ActiveX). DAO מיועד למנגנון מסד הנתונים Jet, ומאפשר תכנות פשוט ומהיר של מסד הנתונים. Access 2000 הוא הגירסה הראשונה של Access שתומכת גם ב-ADO בכל הקשור לטיפול במסדי נתונים תומכי-Jet. במקום להתבסס על מנגנון מסד נתונים יחיד, ADO מנצל מודל תכנות משותף כדי לספק גישה לנתונים אוניברסליים, כשאת הקישורים הבסיסיים למקורות נתונים הוא מקבל מספקי OLE DB. טכנולוגיות OLE DB תדחקנה בסופו של דבר את טכנולוגיות ODBC שקדמו להן, כפי שמודל ADO יחליף את מודל DAO. אם תלמד לעבוד עם ADO כבר עתה, תוכל לאמץ במהירות את השיפורים העתידיים בתחום הגישה לנתונים של גרסאות Access שתבואנה בהמשך, כולל האפשרות לנצל מקורות נתונים נוספים ושונים.

בפרק זה נסקור את מודלי הגישה לנתונים DAO ו-ADO, תוך שימת דגש מיוחד על ADO כמודל תכנות. הסקירה התמציתית על DAO מציגה את תפיסות הפיתוח העיקריות שלו ומבט היסטורי על גישה לנתונים באמצעות Access. מודל DAO לא ימלא תפקיד מרכזי בפרקים הבאים, ולכן בפרק זה נציג את השימוש ב-Jet ובמסדי נתונים מרוחקים. לקבלת מידע על קוד DAO, עיין בעזרה המקוונת של Access ובקר באתר התמיכה המקוונת של Microsoft (support.microsoft.com/support). האתר מתעד בעיות טיפוסיות ודרכי התגברות עליהן. מאמרים רבים באתר כוללים דוגמאות קוד.

בפרק הנוכחי נתמקד בעיקר במודלי האובייקט ADO עבור Jet ביישום Access ובספריות ADODB ו-ADOX. נציג דוגמאות קוד מקיפות שממחישות כיצד לבצע מטלות מסד נתונים אופייניות. הפרקים האחרים בספר מתבססים על הפרק הנוכחי ועוסקים בהיבטי ADO נוספים, כגון שכפול מסד נתונים, גישה למסד נתונים מרוחק ואבטחה בסביבה רבת-משתמשים.

סקירה כללית על DAO

Access 2000 כולל את גרסת 3.6 של ספריית DAO, שהיא גרסת שדרוג תחזוקה של גרסת 3.5 ששווקה עם Access 97 (קיים דמיון רב בין הארכיטקטורה הבסיסית והרכיבים התפקודיים של שתי הגרסאות). DAO מתבסס על מודל אובייקט סביבת העבודה לסוגי גישה לנתונים. אובייקט סביבת העבודה יכול להכיל מידע הפעלה (session), אבטחה וטרנזקציה (אובייקט סביבת עבודה מגדיר את האינטראקציה בין היישום והנתונים).

קיימים שני סוגי סביבת עבודה: סביבות עבודה של **Microsoft Jet** וסביבות עבודה של **ODBCDirect**.

סביבות העבודה של Jet

סביבות העבודה של Jet (Jet workspaces) מיועדות למקורות נתונים Jet, ODBC הקשור ל-Jet ומקורות נתונים ISAM הניתנים להתקנה. מקורות נתוני ODBC הקשור ל-Jet מאפשרים להתחבר אל מקורות נתונים מרוחקים בסביבת DAO המוכרת. למרבה הצער, סוג חיבור זה מחייב שימוש במודל DAO במלואו, והוא טוען את Jet גם כשאין צורך בגישה לנתונים. מקורות נתונים ISAM הניתנים להתקנה, משווקים במיגוון תבניות, כגון Paradox ו- Lotus 1-2-3.

לסביבות העבודה המקובלות של Jet יתרונות אחדים:

- עדכון נתונים באובייקטים של ערכת רשומות,
- צירוף טבלאות ממקורות נתונים שונים לערכת רשומות אחת,
- יצירת טבלאות המבוססות על שיטות DAO מוכרות, במקום על מוסכמות שפת DDL (Data Definition Language) של SQL,
- איגוד נתונים לטפסים ולדוחות.

לסביבות העבודה של Jet אובייקטים של אוספי Groups (קבוצות) ו-Users (משתמשים), סביבות העבודה של ODBCDirect אינן כוללות אובייקטים כאלה. הסיבה היא שמקורות מסדי נתונים מרוחקים, כגון Microsoft SQL Server יכולים לנהל בעצמם את האבטחה שלהם.

סביבות העבודה של ODBCDirect

סביבות העבודה של ODBCDirect (ODBCDirect workspaces) היא טכנולוגיית DAO חדשה יחסית שהוצגה לראשונה ב- Access 97 וב- DAO 3.5. משום שבעבודה עם מקורות נתונים מרוחקים של ODBC אין שימוש באובייקטי האבטחה של DAO, ואובייקטי DAO אחרים פועלים בצורה הטובה ביותר עם מקורות נתונים שמחוברים בצורה מקומית, Microsoft יצרה את מודל האובייקט ODBCDirect, הזמין מסוג סביבת עבודה נפרד. סביבת עבודה זו מאפשרת גישה מהירה וישירה אל מקורות נתונים מסוג

ODBC (כגון SQL Server) תוך עקיפת הצורך להשתמש ב-Jet. אינך מאבד את האפשרויות הרבות הטמונות במודל אובייקט, ואינך חייב להסתמך באופן בלעדי על פקודות SQL כמו במקרה של שאילתות מעבר של SQL.

להלן מספר היתרונות העיקריים של סביבת העבודה ODBCDirect :

- שימוש במקורות נתונים מרוחקים ללא טעינת מנגנון Jet,
- שאילתות אסינכרוניות,
- גישה משופרת לתפקודיות מסד נתונים מרוחק, כולל סמנים ושגרות מאוחסנות (Stored Procedures),
- עדכון אצוות של מקורות מרוחקים מתוך מטמון מקומי,
- החזרת ערכות תוצאות מרובות באמצעות שאילתה בודדת.

לסביבות העבודה של ODBCDirect ספריית סמנים עשירה יותר מזו של סביבות העבודה של Jet, והן תומכות בסמנים דינמיים (dynamic cursors) ובסמני עדכון אצווה (batch update cursors) שאינם זמינים בסביבות עבודה של Jet. הסמן הדינמי מאפשר להפעלה (session) להציג שינויים שנערכו על ידי משתמשים אחרים, מבלי לבצע מחדש שאילתה על מקור הנתונים. סמן עדכון האצווה מאפשר לעדכן אסינכרונית מקור נתונים חיצוני, דבר התורם לשיפור הביצועים, משום שאין צורך לנעול רשומות.



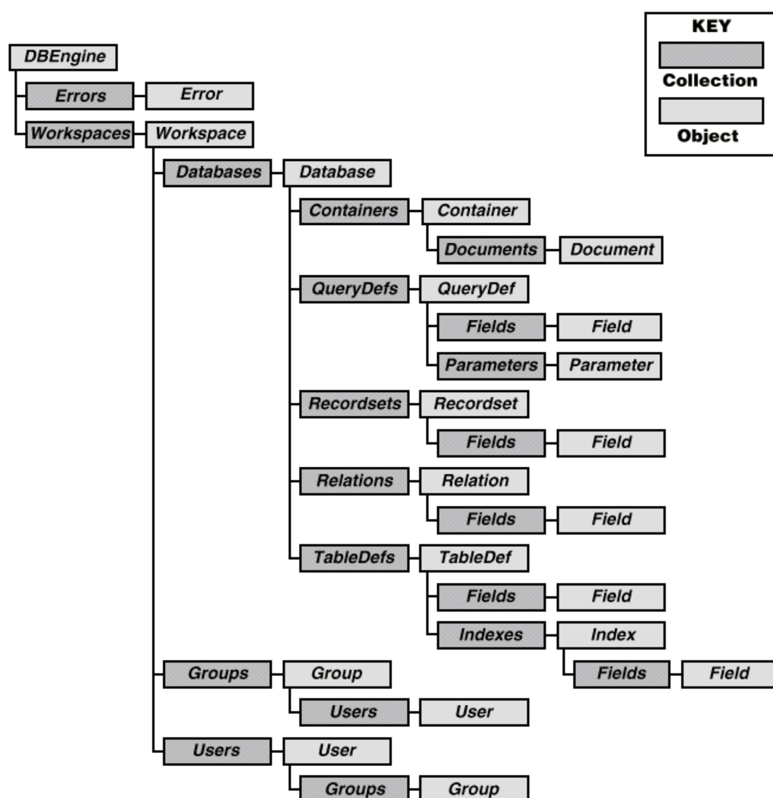
הערה:

סמנים מגדירים את סוג ומיקום הגישה למקור נתונים. הגדרת סמן עשויה לכלול מספר מאפייני מקור נתונים. לדוגמה, סמן יכול להיות בר-עדכון או לא. סמן יכול לאפשר תנועה קדימה בלבד או תנועה דו-כיוונית. הם יכולים להתעדכן באופן אוטומטי כדי לשקף שינויים או הודעות במסד נתונים, והם יכולים לחייב פעולת רענון מפורשת כדי להציג גירסה עדכנית ביותר של מסד הנתונים. תוכל לייעד סמן שפועל בשרת מסד נתונים מרוחק, או בתחנת עבודה מקומית.

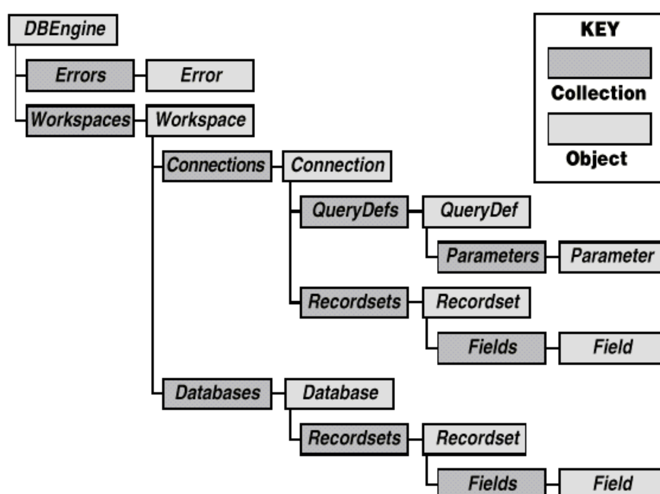
כדי להפיק את מירב התועלת משתי סביבות העבודה, עבוד עם סביבות עבודה מרובות של שני הסוגים כשהן פתוחות בו-זמנית. פעולה זו מאפשרת לנצל היטב את פשטות הטפסים המאוגדים לנתונים שמקורם בסביבת עבודה של ODBCDirect. כל שעליך לעשות לשם כך הוא להשתמש ברשומות המוחזרות בסביבת עבודה של Jet.

אובייקטים המשותפים לסביבות העבודה של Jet ו-ODBCDirect

בשני סוגי סביבות העבודה, אובייקטים של DAO מאורגנים בדרך כלל בצורה היררכית. תרשים 2.1 מציג את ההיררכיה של האוספים והאובייקטים של DAO בסביבות עבודה של Jet. תרשים 2.2 מציג את ההיררכיה של האוספים והאובייקטים של DAO בסביבות עבודה של ODBCDirect.



תרשים 2.1: אוספים ואובייקטים של DAO בסביבות עבודה של Jet



תרשים 2.2: אוספים ואובייקטים של DAO בסביבות עבודה של ODBCDirect

האובייקט DBEngine

DBEngine הוא אובייקט DAO ברמה הגבוהה ביותר המיועד לשתי סביבות העבודה. השיטה **CreateWorkspace** (יצירת סביבת עבודה) של אובייקט זה, משמשת לפתיחת **הפעלה** (session). Type (סוג) הוא ארגומנט אופציונלי של השיטה, שמאפשר להגדיר סביבת עבודה של Jet או סביבת עבודה של ODBCDirect. תוכל גם לקבוע את המאפיין DefaultType (סוג ברירת מחדל) של האובייקט DBEngine כך שאחת מסביבות העבודה תיפתח במקרה של העדר הגדרה מסוימת לשיטה **CreateWorkspace**. סביבת העבודה מסוג Jet היא ברירת המחדל הטבעית. הגדרה כלשהי של הארגומנט Type השייך לשיטה CreateWorkspace תדרוס את המאפיין DefaultType, בין אם הוגדר מפורשות, ובין אם לא.

המאפיינים והשיטות של DBEngine הזמינים לשני סוגי סביבות העבודה כוללים קבוצת פונקציות DAO בסיסיות. השתמש ב-DBEngine ליצירת מסדי נתונים וניהולם.

השיטה **CreateDatabase** (יצירת מסד נתונים) יוצרת סביבות עבודה חדשות. תוכל להפעילה עם ארגומנטים כדי להגדיר את סדר המיון ומצב ההצפנה של מסד הנתונים. תוכל גם להגדיר את תבנית הגירסה של מסד נתונים כדי ליצור מסדי נתונים בצורה מתוכנתת, התואמים לגרסאות קודמות של Access. השיטה **OpenDatabase** (פתיחת מסד נתונים) פותחת מסד נתונים נוכחי באובייקט של סביבת עבודה. לאחר יצירת מסד הנתונים תוכל להחיל עליו את השיטות **Compact** (דחיסה) ו-**Repair** (תיקון) כדי לנהל אותו.

DAO מאפשר לעבד **טרנזקציות** (Transaction) באמצעות שלוש שיטות של האובייקט DBEngine : **BeginTrans**, **CommitTrans** ו-**RollBack**. טרנזקציה היא קבוצת פעולות שמתבצעות בצורת הכל-או-לא-כלום. אם נכשל אחד השלבים בשרשרת פעולות מסד נתונים, ניתן לבטל את כל הפעולות (Roll back). לדוגמה, אם הבנק מעביר סכום כסף מחשבון עו״ש לתוכנית חיסכון, שתי הפעולות יחד - חיוב העו״ש וזיכוי חשבון החיסכון - חייבות להסתיים בהצלחה כדי שהרישום בספרים יתאזן. אם אחת הפעולות אינה מסתיימת בהצלחה, יש לשחזר את המצב שקדם לפעולות בשני החשבונות. טרנזקציות יכולות להאיץ את תהליכי העיבוד במסדי הנתונים על ידי ארגון פעולות הכתיבה לדיסק באצוות. ניתן לקנן טרנזקציות עד חמש רמות.

חלק מהשיטות והמאפיינים הזמינים תלויים בסוג סביבת העבודה, אך בכל סביבת עבודה, תהא אשר תהיה, DBEngine יכלול את האוספים Errors (שגיאות) ו-Workspaces (סביבות עבודה).

האוסף Errors

האוסף **Errors** משמש לטיפול בשגיאות המתרחשות בעת גישה לנתונים. האוסף מכיל מספרי שגיאה ותיאור המשפט האחרון שנכשל. בעבודה עם מקורות נתונים מסוג ODBC אפשר לקבל שגיאות רבות, מכיון ששכבות שונות של ODBC עשויות לדווח על שגיאות שמקורן באותה תקלה, כגון מקור מסד נתונים מרוחק שאינו פעיל.

האוסף Errors, בדומה לאוספי DAO אחרים, הוא מבוסס-אפס. המאפיין Count (מונה) מציין את מספר השגיאות באוסף, וכל שגיאה נושאת מספר מ-0 ועד Count –1. הערך האחרון באוסף Errors מקביל לאובייקט Err של Microsoft Visual Basic for Applications (VBA).

במהלך ניפוי הבאגים מהיישום, ייתכן שתגלה כי כדי לפשט את החיפוש אחר מקור השגיאה או פתרון הבעיה, כדאי למספר את האוסף.

האוסף Workspace

המאפיינים והשיטות של האוסף **Workspace** משמשים כדי להפנות להפעלות סביבות עבודה בודדות. האוסף Workspace זמין תמיד, Access יכול לקיים הפעלות רבות בו-זמנית, וגם היישומים יכולים לפתוח ולנהל צירופים שונים של סביבות העבודה Jet ו-ODBCDirect. הפעלות אלו אינן מתקיימות מעבר להפעלת הכניסה, אך הן מסוגלות להתקיים לכל אורך ההפעלה. באמצעות אחת התבניות הבאות תוכל לנצל את הארגומנט Name (שם) של השיטה **CreateWorkspace** כדי להפנות בצורה ייחודית לסביבות עבודה מסוימות באוסף:

- DBEngine.Workspaces(0)
- DBEngine.Workspaces("Name")
- DBEngine.Workspaces![Name]

כל אובייקטי DAO מבוססים על תחביר הפניה זהה לתבניות שבדוגמה. שני הסגנונות הראשונים עומדים בכללי ADO. בתחילת דרכך בתכנות ADO, עליך לאמץ אחת מהן כדי לשפר את מיומנויות ההגירה שלך.

אובייקטי Workspace משתפים שיטות חשובות אחדות עם האובייקט DBEngine, כגון CreateDatabase, OpenDatabase ו-BeginTrans. שיטות נוספות כגון CreateUser ו-CreateGroup הן ייחודיות לאובייקט Workspace. שתי השיטות האלו מסייעות לנהל אבטחה ברמת-משתמש בתוך סביבת העבודה.

האוסף Database

באמצעות קוד אפשר לפתוח מסדי נתונים רבים מתוך סביבת עבודה כלשהי. השימוש ב-DAO בשילוב עם VBA מעניק יתרון ברור על פני ממשק המשתמש שמאפשר פתיחת מסד נתונים יחיד בזמן נתון. ניתן להאיץ את הגישה על ידי שימוש בשמות משתנים לצורך הפניה לאובייקטים של מסד נתונים. גישה זו משפרת את מהירות הגישה לאובייקטי DAO רבים.

המאפיין Name של מסד נתונים בודד בסביבת עבודה של Jet, הוא הנתיב אל קובץ מסד הנתונים. קוד VBA מתייחס למסד הנתונים בפרויקט הנוכחי באמצעות הפונקציה **CurrentDb**. Access תומך גם בתחביר חלופי של: DBEngine (0)(0) כדי להפנות למסד הנתונים הנוכחי. תחביר CurrentDb יוצר מופע נוסף של פרויקט מסד הנתונים הנוכחי, אך תחביר DBEngine מתייחס לעותק הפתוח של מסד הנתונים הנוכחי. תוכל לפתוח

אובייקטים של מסד נתונים במקורות נתונים השונים מ-Jet, כגון ISAM (לדוגמה, dBase או Microsoft FoxPro). בצורה זו ניתן להשתמש במקור נתונים ODBC, אך הביצועים ישתפרו כשמשתמשים בסביבת העבודה ODBCDirect (כפי שיתואר בסעיף **אובייקטי סביבת העבודה של ODBCDirect**).

האוסף Recordset

אובייקט **Recordset** (ערכת רשומות) מייצג רשומות בטבלה או את אלו שהתקבלו כתוצאה משאילתת החזרת שורות (row-returning). ערכת רשומות חדשה נוצרת באמצעות השיטה **OpenRecordset**. אפשר להפעיל את השיטה מתוך אובייקטים אחדים, לרבות אובייקטי מסד נתונים ו-TableDef, וליצור בעזרתה ערכת רשומות. עם האובייקטים הנוספים הכוללים את השיטה OpenRecordset, נמנים אובייקטי QueryDef ואפילו ערכות רשומות נוספות. שיטה זו מוסיפה ערכת רשומות חדשה לאוסף ערכות הרשומות. אפשר ליצור 5 סוגי ערכות רשומות, כמתואר בטבלה הבאה.

אובייקטים של ערכת רשומות

סוג	תיאור
Table	סוג זה מתייחס לרשומה בטבלה, כגון זו שיוצרים באמצעות השיטה CreateTableDef. הסוג מתייחס רק לטבלה יחידה של Jet. ניתן לעדכן ערכי שדה, להוסיף ולמחוק רשומות. אין סמן ODBC מקביל
Dynaset	סוג זה הוא אוסף רשומות דינמי שעשוי להיות תוצאה של טבלה אחת או יותר. השדות הנבחרים ניתנים לעדכון, ולכן ניתן להוסיף, למחוק ולשנות רשומות. המאפיין DataUpdatable משמש לקביעה אם שדה כלשהו הוא בר-עדכון. במסד נתונים מרובה-משתמשים, ניתן להציג שינויים נבחרים שנערכו על ידי משתמשים אחרים. סוג ערכת רשומות זה מקביל לסמן keyset של ODBC
Snapshot	סוג זה מאפשר לבחון רשומות המבוססות על טבלה אחת או יותר, אך אינו מאפשר לשנות אותן. ברגע שטוענים תמונה (Snapshot) לזיכרון, היא אינה משקפת שינויים נוספים כלשהם שנערכו בטבלאות. סוג ערכת רשומות זה מקביל לסמן static של ODBC
Forward-only	סוג זה זהה עקרונית לערכת הרשומות Snapshot, אך ניתן לגלול אותו קדימה בלבד. סוג זה מקביל לסמן forward-only של ODBC
Dynamic	סוג זה מייצג תוצאת שאילתה המבוססת על טבלה אחת או יותר. משתמשים יכולים לעדכן את ערכת הרשומות על ידי הוספה, מחיקה ושינוי רשומות. סוג זה גם מציג שינויים שנערכו על ידי משתמשים נוספים בסביבה מרובת-משתמשים. סוג זה זמין רק בסביבות עבודה של ODBCDirect ומקביל לסמן dynamic של ODBC

השיטה OpenRecordset. שיטה זו זקוקה לארגומנט מקור שמציין את מקור תוצאת ערכת הרשומות. כפי שהוזכר, ארגומנט טיפוסי הוא שם טבלה, שם שאילתה או משפט SQL. ניתן לציין גם את סוג ערכת הרשומות; אם אין מציינים, DAO מחזיר ערכת רשומות מסוג table, dynaset או forward-only, בהתאם למקור הנתונים.

ארגומנט אפשרויות (options argument) מאפשר לציין אחת ממספר תכונות של ערכת רשומות, כגון מניעת כתיבה לערכת רשומות או קריאה ממנה. ארגומנט סופי (final argument) מאפשר לקבוע את המאפיין LockEdits של ערכת רשומות. מאפיין זה מציין את סוג הנעילה התקפה בעת שיישום מעדכן, מוסיף או מוחק רשומות מתוך ערכת רשומות. בסביבות עבודה של Jet, ההגדרות שניתן לספק לארגומנט זה תהיינה בדרך כלל read-only, pessimistic locks ו- optimistic locks. אם ההגדרה היא read-only, לא ניתן לעדכן את ערכת הרשומות. pessimistic locks גורם לשיטה Edit לנעול את הדף שמכיל את הרשומה. במצב optimistic locks, משתמשים אחרים יכולים לעדכן את הרשומה עד שהיישום מפעיל את השיטה Update. סידור זה יכול לשפר את ביצועי ערכת הרשומות, אך עלול גם לגרום להתנגשויות בין פעולות עדכון.

שיטות Recordset. אפשר לשנות את הרשומות בערכת רשומות באמצעות השיטות **Edit**, **AddNew**, **Update** ו-**Delete**. השיטות Edit ו-Update משמשות במשולב לעדכון הערכים בערכת רשומות. השיטה Edit פותחת רשומה לצורך עריכה, והשיטה Update מחילה את הערכים החדשים על הטבלאות שבבסיס ערכת הרשומות. השיטות AddNew ו-Update פועלות במשולב. הקוד מודיע על הכוונה להוסיף רשומה על ידי הפעלת השיטה AddNew. לאחר מכן שומרים את הרשומה החדשה באמצעות השיטה Update. השיטה Delete מסירה את הרשומה הנוכחית מערכת הרשומות. לאחר שרשומה נמחקת היא ממשיכה להתקיים עד למעבר לרשומה חדשה. השיטה Delete אינה מחייבת את השימוש בשיטה Update.

בעת הוספת ערכת רשומות לאוסף Recordset אפשר להשתמש בקבוצת השיטות **Move** (מעבר אל) כדי לנווט אותה. השיטה **MoveNext** מנווטת אל הרשומה הבאה, והשיטה **MovePrevious** מנווטת אל הרשומה הקודמת. כשנמצאים ברשומה הראשונה ומפעילים את השיטה MovePrevious, DAO מחזיר סמן **BOF** (Beginning Of File) מערכת הרשומות. אפשר לנצל סמן זה כדי לציין מעבר לרשומה כלשהי שבאה לאחר הרשומה הראשונה. באופן דומה, כשנמצאים ברשומה האחרונה ומפעילים את השיטה MoveNext, DAO מחזיר סמן **EOF** (End Of File). ניסיון לנוע מעבר לסמנים BOF או EOF יוצר שגיאת זמן ריצה (Run-Time Error). השיטה Move מאפשרת לציין מספר שורות קבוע למעבר. אפשר גם לקבוע מיקום התחלתי שונה מהרשומה הנוכחית. השיטות **MoveFirst** ו-**MoveLast** עוברות ישירות לרשומה הראשונה או האחרונה בערכת הרשומות, בהתאמה. בערכות רשומות גדולות מאוד, עשויה לחול השהיה משמעותית במהלך ההגעה לרשומות הקצה שבערכת הרשומות.

קבוצת שיטות נוספת מנווטת אל רשומה חדשה שעונה על קריטריונים מוגדרים. שיטות אלו הן **FindFirst**, **FindLast**, **FindNext** ו-**FindPrevious**. קריטריוני השיטה מסוג Find (חיפוש) נקבעים באמצעות תחביר זהה לזה של הפסוקית WHERE שבמשפט SQL. אם לא קיימת ערכת רשומות שעונה על הקריטריונים שצוינו, שיטות אלו

תספקנה למאפיין NoMatch של ערכת הרשומות את הערך True. אחרת, הן פשוט תעבורנה אל הרשומה שעונה על הקריטריונים. השיטות FindNext ו-FindPrevious מתחילות את החיפוש מהרשומה הנוכחית. השיטות FindFirst ו-FindLast עורכות חיפוש מהרשומה הראשונה או האחרונה, בהתאמה. בטיפול באובייקטים מסוג TableDef, השיטה Seek מסוגלת להפיק תוצאות מהירות יותר מאלו שיכולות לספק שיטות Find. כללית, בעת הפעלת השיטות OpenRecordset ניתן לקבל תוצאות חיפוש טובות יותר באמצעות משפטי SQL. בשימוש בשיטות Find, Move ו-Seek, רצוי להגדיר את המאפיין Index של ערכת רשומות, כדי לאפשר ארגון הרשומות בסדר הנכון שנקבע על ידי שדות האינדקס.

אובייקטי סביבת העבודה של Jet

אובייקטים של מסד נתונים בסביבות העבודה של Jet כוללים אלמנטים נבחרים של סכימת מסד נתונים, ובאפשרותם גם להפעיל אותם. לדוגמה, באפשרותך לפתוח ערכות רשומות לטיפול או להפעיל שאילתות פעולה שמעדכנות, מצרפות או מוחקות רשומות. שיטות של מסד נתונים אף מאפשרות ליצור ולנהל עותקים. לפניך חמשת האוספים ההירככיים של מסד נתונים. האובייקט Database כולל שיטות להוספת אלמנטים חדשים לכל האוספים הללו.

- TableDefs
- Recordsets
- QueryDefs
- Relations
- Containers

האוסף TableDefs

אוסף זה מאפשר לגשת לאובייקטים בודדים של TableDef במסד נתונים. אובייקטים אלה מכילים את האוספים Fields ו-Indexes, ולכן הם מאפשרים להגדיר טבלה באמצעות אובייקט TableDef. באפשרותך להשתמש בשיטות CreateField ו-CreateIndex כדי להרכיב את הגדרת הטבלה. בעת השימוש בשיטה CreateField, מגדירים תחילה את השדה על ידי ציון שמו, סוגו וגודלו. לאחר מכן מפעילים את השיטה Append כדי להוסיף את השדה החדש לאוסף Field עבור אובייקט TableDef. אם אוסף מכיל כבר שדה ששמו צוין כארגומנט, הקוד שכתבת ייצור שגיאת זמן ריצה שניתן ללכוד אותה. תוכל לנצל זאת לניהול האובייקט TableDef למשל, על ידי הסרת השדה הישן באמצעות השיטה Delete.

בעת יצירת אינדקסים, מפעילים את השיטה CreateIndex ומצרפים לאינדקס שדה אחד או יותר. לאחר מכן מוסיפים את האינדקס החדש לאוסף Indexes עבור האובייקט TableDef. אם קיים כבר אינדקס בשם זה, תקבל שגיאת זמן ריצה. תוכל לנצל שגיאות אלו כדי לנהל את תהליך יצירת האינדקס של האובייקט TableDef.

האובייקט TableDef יכול לנהל קישורים אל טבלאות במקורות הנתונים ISAM ו-ODBC. הקוד דורש את המאפיינים Connect ו-SourceTableName ואת השיטה CreateTableDef. השיטה CreateTableDef מופעלת כדי להגדיר הפניה במשתנה לטבלה המקושרת. אחר כך מגדירים את המאפיינים Connect ו-SourceTableName של המשתנה. המאפיין Connect מציין את סוג מקור הנתונים, כגון dBase 5.0 או Paradox 5.x, ואת הנתיב אל מקור הנתונים הספציפי שאליו תיצור את הקישור. המאפיין SourceTableName הוא שם הטבלה המקושרת. לאחר קביעת מאפיינים אלה, משלימים את התהליך על ידי צירוף האובייקט TableDef לאוסף TableDefs.

האוסף QueryDefs

אוסף זה מאחסן את האובייקטים הבודדים מסוג QueryDef במסד נתונים. אובייקט QueryDef הוא משפט SQL שמחזיר באופן טיפוסי קבוצת שורות או מבצע פעולה, כגון עדכון, הוספה או מחיקת רשומות בערכת רשומות. כשמשפט SQL של אובייקט QueryDef מחזיר שורות, יכול להיות לו אוסף Fields ובו שדות יחידים. אובייקט QueryDef יכול להכיל אוסף Parameters, אם משפט SQL שכתבת מקבל ארגומנטים שמציינים את הקריטריונים שלו בזמן ריצה. כשהאובייקט QueryDef פועל, באפשרותך לציין פרמטרים אלה בצורה מתוכנתת (באמצעות קוד), או לאפשר למשתמש לעשות זאת בזמן ריצה באמצעות תיבת דו-שיח.

תוכל ליצור אובייקטים חדשים מסוג QueryDef באמצעות השיטה CreateQueryDef עבור האובייקט DataBase בסביבת העבודה של Jet, או האובייקטים DataBase או Connection בסביבת העבודה של ODBCDirect. אם תיתן לאובייקט QueryDef שם באמצעות מחרוזת שאורכה שונה מאפס, DAO יוסיף את השם אוטומטית לאוסף QueryDefs וישמור את האובייקט QueryDef באופן קבוע בדיסק יחד עם מסד הנתונים. השתמש בשיטה Delete כדי להסיר פריט מהאוסף QueryDefs. כל אובייקט QueryDef שהמאפיין Name שלו הוא מחרוזת באורך אפס, הוא אובייקט זמני, ו-DAO אינו מקיים אותו לאורך זמן. אובייקטים QueryDef זמניים נוחים לשימוש כשהיישומים צריכים ליצור אובייקטים כאלה באופן דינמי.

שתי שיטות מאפשרות להפעיל אובייקט QueryDef. השיטה OpenRecordset מחזירה את השורות שבאובייקט QueryDef באמצעות המשפט SELECT, והשיטה Execute מבצעת שאילתת פעולה. האפשרות dbFailOnError יכולה לאפשר ליישום לקבוע אם אובייקט QueryDef נכשל בביצוע פעולתו המיועדת על כל הרשומות העומדות בדרישות הקריטריונים. כל עוד אובייקט QueryDef מציית לכללי התחביר, הוא לא יגרום לשגיאה, גם במקרה שייכשל בביצוע פעולתו. האפשרות dbFailOnError מחזירה את המצב לקדמותו (roll back), אם האובייקט QueryDef אינו מסוגל לבצע את כל השינויים. אפשרות זו אף יוצרת שגיאת זמן-ריצה כדי לסייע לבצע את פעולות העיבוד הקשורות בתהליך, כגון הצגת הודעות למשתמש.

האוסף Relations

האוסף Relations (קשרי גומלין) ומערכת קשרי הגומלין שלו משמשים להגדרת קישורים (links) בין טבלאות באמצעות קוד. השיטה CreateRelation של האובייקט Database יכולה לאתחל קשרי גומלין; היא מאפשרת ליישום להגדיר קשרי גומלין מסוג יחיד-ליחיד או יחיד-לרבים בין שתי טבלאות, לקבוע שלמות קשרים (referential integrity) ולארגן מחיקות ועדכונים על פי היררכיית קשרים (פרק 4 מרחיב בנושא). מגדירים קשרי גומלין בין טבלאות בהתאם לשדות משותפים בשתי הטבלאות. האובייקט Relation כולל אוסף Fields התומך בפונקציה זו. האוסף Relations ואובייקטים בודדים מסוג Relation ייחודיים לסביבות העבודה של Jet. הם אינם זמינים בסביבות העבודה של ODBCDirect, מכיון שמנגנוני מסדי נתונים מרוחקים שומרים באופן טיפוסי בעצמם על קשרי גומלין בין טבלאות.

האוסף Containers

אוסף זה מגדיר קבוצת אובייקטים של מכולה עבור מסמכי מסד נתונים. חלק מהאובייקטים מגיעים מחלון **מסד נתונים** (Database): **טפסים**, **דוחות**, **Scripts של מאקרו** ו**מודולים**. כל אלה הם אובייקטים של Access ולא אובייקטים של מסד הנתונים Jet. האובייקטים הנוספים של המכולה הם אובייקטים מבוססי-Jet: **מסדי נתונים**, **טבלאות** ו**קשרי גומלין**. אובייקט המכולה **טבלאות** כולל מידע על טבלאות ושאילתות כאחד. אובייקט מכולה נוסף מכיל נתוני מתאר של קשרי גומלין שנשמרו.

אובייקטי מכולה מאחסנים מסמכים המבוססים על כל הרכיבים השמורים של סוג, כגון טפסים וקשרי גומלין. מסמכים אלה מספקים מידע מינהלי (ולא תוכן) על האובייקטים שבאובייקט מכולה. מאפיינים נבחרים של מסמכים כוללים תאריכי יצירה ועדכון של המסמך ואת שם בעליו והרשאות הטיפול בו. Jet משתמש במידע הכלול במסמך לניהול אבטחת האובייקטים של Access, ושל הטבלאות והשאילתות שלו עצמו.

חשוב להבין כי מסמכים באובייקטי מכולה שונים מאלמנטים של אוסף. מסמכים מכילים אובייקטים שמורים – בין אם הם פתוחים ובין אם לאו. אוספים הם קבוצות אובייקטים פתוחים. אם אובייקט אינו פתוח, הוא אינו מהווה חלק מאוסף, אך הוא עשוי להשתייך לאובייקט מכולה. בנוסף, מסמכים מכילים מידע מינהלי על אובייקטים, אך אלמנטים באוסף מכילים מידע על תוכן, עיצוב ואלמנטים משניים של האובייקטים שבאוסף.

האוספים Users (משתמשים) ו-Groups (קבוצות) מכילים אובייקטים תואמים משלהם. אוספים ואובייקטים אלה משלימים את מסמכי המכולה כדי לסייע ל-Jet לנהל אבטחה ברמת-משתמש. מסמכים כוללים הרשאות. משתמשים שייכים לקבוצות. הרשאות מסמך מתארות רמות גישה של משתמשים וקבוצות. פרק 10 עוסק בהרחבה באבטחה ברמת-משתמש, לרבות משתמשים, קבוצות והרשאות.

אובייקטי סביבת העבודה של ODBCDirect

כשמשווים את תרשים 2.2 לתרשים 2.1, ניתן להבחין כי מודל ODBCDirect חסכן יותר, מכיון שסביבות העבודה של ODBCDirect מעבירות לשרתי מסדי נתונים מרוחקים חלק מהפונקציות שמנהל Jet. לדוגמה, שרתי מסדי נתונים מרוחקים מנהלים את האבטחה בעצמם, ולכן אין צורך באוספים Users ו-Groups. בנוסף, לא קיים אוסף TableDefs, מכיון ששרתי מסדי נתונים מרוחקים מנהלים את טבלאותיהם בעצמם. כך הדבר גם לגבי קשרים.

ישנם הבדלים נוספים בין שני מודלי סביבות העבודה: מודל ODBCDirect כולל אוסף Connections חדש בעל אובייקטים מקבילים משלו. אובייקט Database ממלא תפקיד אחר במודל ODBCDirect, ואובייקטי QueryDef אינם מכילים אוסף Fields. ניתן לגזור אובייקט Recordset מאובייקט QueryDef באמצעות השיטה OpenRecordset.

למרות ששתי סביבות העבודה כוללות אובייקט מסוג Database, התנהגות האובייקט בסביבת העבודה של ODBCDirect שונה במידת-מה מהתנהגותו בסביבת העבודה של Jet. בסביבת העבודה של ODBCDirect, אובייקט זה מכיל את המאפיין Count שמחזיר הפניה אל Connection. האובייקט Connection מכיל את המאפיין Database שמחזיר הפניה אל אובייקט Database. במודלי DAO, האובייקטים Connection ו-Database מהווים דרכים שונות להפנות למשהו זהה. מאפיינים אלה מפשטים את המעבר ממודלי סביבת העבודה של Jet אל סביבות העבודה של ODBCDirect וחזרה.

האוסף Connections

האוסף Connections של סביבת העבודה והאובייקטים שלו מהווים גורמים קריטיים בעת טיפול במסדי נתונים מרוחקים. בסביבות העבודה של ODBCDirect מנצלים את השיטה OpenConnection כדי להקים חיבור (connection) אל מסד נתונים מרוחק. הדבר מחייב ארבעה ארגומנטים, ששלושה מהם אופציונליים. הארגומנט הנדרש הוא השם. מתן שם לחיבור מוסיף אותו לאוסף Connections. שלושת הארגומנטים האחרים מגדירים את טבעו של החיבור. מאחר שהם אופציונליים, ניתן להגדירם בעת יצירת החיבור או לאחר מכן. לעומת זאת, יש להגדיר חיבור בטרם ניתן יהיה להשתמש בו לשליפת נתונים לביצוע פעולה אחרת כלשהי במקור הנתונים המרוחק. לארגומנט Connect חשיבות מיוחדת, בהיותו הארגומנט שקובע את המאפיין Connect של האובייקט Connection. ארגומנט זה מגדיר את מחרוזת החיבור. הוא מתחיל ב-ODBC ותו נקודה-פסיק (;) ולאחר מכן באים נתוני החיבור הדרושים לקישור אל מקור הנתונים המרוחק. נתונים אלה יכולים לכלול שם מסד נתונים ושם קישור ב-ODBC והם גם יכולים להכיל שם משתמש וסיסמה תואמת. מפרידים בין סוגי המידע השונים של הארגומנט באמצעות תווי נקודה-פסיק. יישום יכול לבחון ולאתחל את מחרוזת החיבור באמצעות המאפיין Connect של האובייקט Connection.

פרמטר אופציונלי נוסף של השיטה OpenConnection שולט בשני תפקודים נפרדים: אופן התגובה של חיבור למידע לא שלם של מחרוזת חיבור, וצורת הפתיחה האסינכרונית של חיבור. במקרה של מחרוזת חיבור שהמידע שבה אינו שלם, ניתן

לאפשר לחיבור להיכשל וליצור שגיאת זמן ריצה, או ללכוד את השגיאה ולבקש מהמשתמש להשלים את המידע. באפשרותך גם לנצל פרמטר זה כדי לקבוע פתיחה אסינכרונית של חיבור. היישום יכול לפתוח את החיבור ואחר כך לעבור למשימות אחרות. יישום Access יכול לשרת את המשתמש המקומי על ידי פתיחת טפסים או אפילו קיום חילופי תקשורת עם המשתמש. באותו זמן ממש, שרת מסד הנתונים המרוחק מטפל בבקשת הקמת החיבור החדש. המאפיין StillExecuting של האובייקט Connection מאפשר ליישום לדגום את החיבור כדי לקבוע את זמינות השימוש שלו.

האובייקט Connection כולל חמש שיטות:

- **OpenRecordset**. שיטה זו מחזירה קבוצת שורות ממקור נתונים מרוחק. מספר רכיבי השיטה בסביבת עבודה של ODBCDirect רב יותר מזה שבסביבת העבודה של Jet. קרוב לוודאי שההבדל הגדול ביותר ביניהם הוא שבעת הפעלת השיטה בסביבת עבודה של ODBCDirect ניתן לציין יותר ממשפט SQL אחד, וכך יכול משפט OpenRecordset יחיד לספק ערכות רשומות רבות לשימוש מקומי.
- **Close**. שיטה זו סוגרת חיבור פתוח.
- **CreateQueryDef**. גם שיטה זו כוללת רכיבים נוספים בסביבת העבודה של ODBCDirect. מה שברור הוא שאובייקטי QueryDef אינם כוללים שדות. אם ברצונך להציג את השורות שהוחזרו באמצעות אובייקט QueryDef, עליך להפעיל את השיטה OpenRecordset של אובייקט זה. כל האובייקטים מסוג QueryDef בסביבות עבודה של ODBCDirect הם זמניים. במודל האובייקט ODBCDirect לא ניתן ליצור שיגרה מאוחסנת במקור נתונים חיצוני. אובייקטי QueryDef יכולים להשתייך לאובייקטי Connection בלבד. השיטה CreateQueryDef אינה קיימת עבור אובייקטים מסוג Database בסביבת עבודה של ODBCDirect, כפי שקיימת כזו בסביבת עבודה של Jet. באפשרותך לפתוח אובייקט Recordset מתוך אובייקט Database בשתי סביבות העבודה.
- **Execute**. שיטה זו מפעילה שאילתות פעולה, שאילתות פרמטר ושאילתות בחירה. מגדירים את הקבוע dbRunAsync כדי לציין שעל האובייקט QueryDef לפעול בצורה אסינכרונית. בדיוק כמו במקרה השיטה OpenConnection, משתמשים יכולים לבצע משימות במקביל לפעולת האובייקט QueryDef. המאפיין StillExecuting מאפשר ליישום לבדוק את מצב ההשלמה של האובייקט QueryDef.
- **Cancel**. הפעלת השיטה ללא ציון פעולה אסינכרונית תסיים שאילתה אסינכרונית ותחזיר שגיאת זמן ריצה. שחרור משאבים שצרך אובייקט QueryDef מתבצע על ידי שיטת Close, או הגדרה של הפניית האובייקט כ-Nothing.

עדכון באצווה

עדכון באצווה (batch updating) הוא אחד החידושים החזקים בסביבות העבודה של ODBCDirect. הוא מאפשר ליישום לטעון קבוצת רשומות, לעדכן אותן בצורה מקומית, ואחר כך לעדכן את רשומות המקור כאצווה בודדת, במקום לעדכן כל אחת בנפרד.

לעדכון אצוות יתרון ברור על נעילת רשומה בודדת. עקב הסיכוי להתנגשויות, מומלץ לפעול בדרך זו רק במקרים בהם קטנים הסיכויים שמשתמשים רבים יערכו שינויים במסד הנתונים. אך עיבוד באצווה כולל מספר רכיבים מוכללים לטיפול בהתנגשויות. לדוגמה, המאפיין BatchCollisions מחזיר סימניות שמצביעות על התנגשויות בערכת רשומות לאחר טעינתה. אפשר לכפות על מסד נתונים מרוחק להיות תואם לעדכון המבוצע, או לקבל את הערך שבמקור הנתונים המרוחק. שלושה מאפייני Field מאפשרים לבדוק ערך מקורי טרם טעינתו, ערך מעודכן בגרסת ערכת הרשומות המקומית ואת ערך השדה החדש במקור הנתונים המרוחק.

לפניך חמשת השלבים ליישום עדכון באצווה בסביבת עבודה של ODBCDirect :

1. הגדר את המאפיין DefaultCursorDriver של סביבת העבודה בתור dbUserClientBatchCursor.
2. צור אובייקט Connection או אובייקט Database.
3. הפעל את השיטה OpenRecordset עבור האובייקט שיצרת בשלב 2 באמצעות הגדרת dbOptimisticBatch עבור הארגומנט LockEdits.
4. ערוך את השדה בצורה מקומית, לפי הצורך.
5. הפעל את השיטה Update של ערכת הרשומות של שלב 3 באמצעות הגדרת dbUpdateBatch עבור ארגומנט הסוג. אם לא אירעו התנגשויות כלשהן, סיימת בהצלחה. אם קיימות התנגשויות, תזדקק ללוגיקה נוספת כדי ליישב אותן.

סקירה כללית על ADO

Access 2000 תומך בגירסה 2.1 של ADO, הכוללת שלושה מודלים לגישה לנתונים: הספריה ADODB, הספריה ADOX והספריה JRO. חלוקת הגישה לשלוש ספריות מאפשרת לתוכניות שאינן זקוקות לכל שלושת המודלים להשתמש רק בחלקם. מרכיב נוסף באסטרטגיית הגישה לנתונים של Access הוא האמון בספקי OLE DB. ספקים אלה פועלים בשיתוף עם ADO כדי לספק גישה למקורות נתונים מקובלים ולחדשים כאחד, כגון ספריות דואר אלקטרוני. סידור זה מעצים את התכנות של מסדי נתונים.

הספריה ADODB היא ספריה קטנה ופשוטה, שמכילה אובייקטים חיוניים ומספקת את הרכיבים הבסיסיים ליצירת חיבורים, הסרת פקודות ואחזור ערכות רשומות. ספריה זו גם מאפשרת לנווט בערכת רשומות. ניתן להשתמש בה לביצוע מטלות אחזקה בסיסיות, כגון שינוי, הוספה ומחיקה של רשומות. העיצוב הלא היררכי של הספריה מקל על משתמשים מתחילים.

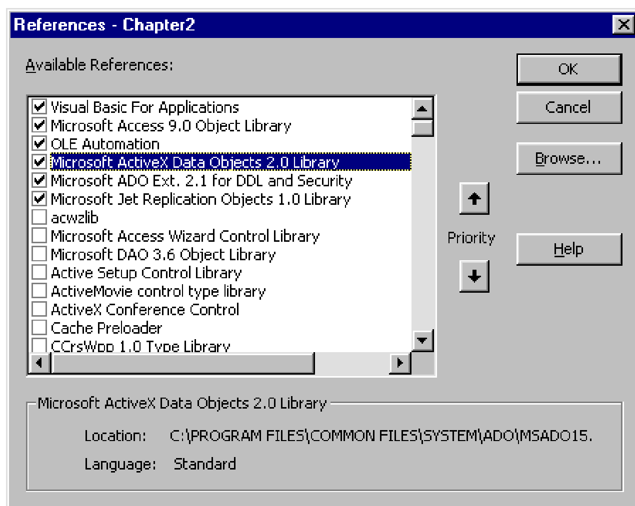
הספריה ADOX תומכת בשפת הגדרת נתונים ובנושאים הקשורים באבטחה. היא כוללת אובייקטים המקשרים את המשתמש עם הסכימה הכוללת של מסד הנתונים. לדוגמה, היא מאפשרת ליצור טבלאות וקשרים. המודל כולל תמיכה בשלמות קשרים (referential integrity), עדכון שדות קשורים ומחיקת רשומות קשורות, ומציע שגרות, תצוגות ואוספים של Users ו-Groups לאבטחת מסד נתונים ברמת-משתמש.

הספריה JRO מאפשרת יצירת שכפול של מסד נתונים Jet. Access 2000 תומך בשכפול מסד נתונים במסדי הנתונים Jet ו- SQL Server. פרק 11 עוסק בהרחבה בשכפול מסד נתונים.

היתרון העיקרי של ADO הוא מודל האירוע. ODBCDirect מאפשר ביצוע פעולות אסינכרוניות, אך ADO גם מספק אירועים. על ידי כך הוא משחרר את היישום מהצורך לבחור אובייקט ומבדיקת המאפיין StillExecuting. במקום זאת, ניתן ליצור מטפלים באירועים שיגיבו על אירועים בעת התרחשותם (בפרקים הבאים נסביר כיצד לבנות מטפלים באירועים).

ספקי OLE DB מסייעים לפתח את העוצמה של ADO. הם מספקים דרך חדשה לגשת לנתונים מרוחקים, דרך שמתבססת על ODBC ומרחיבה את יכולתה, ומציעים גישה למסדי נתונים יחסיים ולמקורות נתונים לא יחסיים באמצעות ממשק ADO עקבי. Access 2000 משווק עם מיגוון של ספקי OLE DB, כולל כאלה עבור SQL Server, Jet, Oracle, מקורות נתוני ODBC כלליים וגם מקורות לא מקובלים, כגון Microsoft Active Directory Service ו- Microsoft Index Server. בבוא הזמן יש לצפות לגידול במספר ספקים אלה.

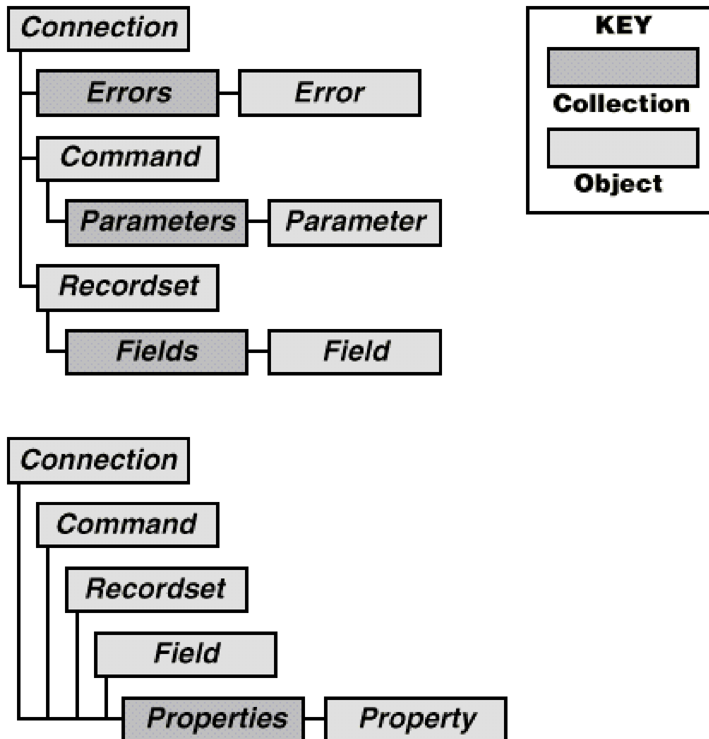
בטרם תוכל להשתמש בספריית ADO כלשהי, עליך ליצור הפניה אל ספריה אחת לפחות. יוצרים הפניה מתוך החלון Visual Basic Editor (VBE) באמצעות הפקודה **References** (הפניות) בתפריט **Tools** (כלים). תרשים 2.3 מציג את תיבת הדו-שיח **References** כשכל שלוש הספריות נבחרו. נוח יותר לבחור את כל השלוש, אך בחירת הספריה הדרושה בלבד משמרת יותר משאבים. בחר לניסיון ספריות שונות ביישומים שונים הפועלים במחשבים שונים, כדי לקבוע את הצירוף המתאים ביותר לסביבת המחשוב. אם ברשותך יישום המשמש לעבודה שוטפת, שפועל בסוגי מחשבים שונים, עליך לשמר משאבים עבור דרישות אחרות של היישום.



תרשים 2.3: תיבת הדו-שיח **References** משמשת להוספת ספריות ADO ליישום

הספריה ADODB

ספריית האובייקטים ADODB כוללת שבעה אובייקטים עיקריים. ארבעה מתוכם כוללים אוספים. האובייקט Connection מופיע בראש המבנה ההירככי, אך ניתן ליצור חיבורים באופן משתמע (implicit) באמצעות אובייקטים אחרים. האובייקטים Connection, Command, Recordset ו-Field כוללים אוספי Properties.



תרשים 2.4: ספריית האובייקטים ADODB

האובייקט Connection

אובייקט זה יוצר קישור אל מסד נתונים. משתמשים באובייקט Connection בצורה משתמעת או מפורשת (explicit) בעת טיפול במסד נתונים. בעת יצירה מפורשת של קישור, ניתן לנהל ביעילות חיבור אחד או יותר ולהקצות מחדש את תפקידיהם ביישום. בעת יצירה משתמעת של חיבור ניתן לקצר את הקוד. כל אובייקט חדש שנוצר באמצעות חיבור משתמע, צורך יותר משאבים. אם היישום מכיל אובייקט אחד או שניים לכל היותר, שכל אחד מהם דורש חיבור נפרד, חיבורים משתמעים ייטיבו לענות על צרכיך. ADO מאפשרת לבחור את דרך היצירה והניהול של חיבורים באופן שתמצא לנחוץ.

בניגוד ל-DAO, ADO היא שפה כללית לגישה לנתונים, ולכן לא כל השיטות והמאפיינים שלה מתאימים למנגנון Jet. לעומת זאת קיים ספק OLE DB עבור Jet 4, הגירסה העדכנית ביותר של Jet המשווקת עם Access 2000. האובייקטים Connection תלויים במידה מכרעת במפרטי הספק, ולכן יש ערך רב ליכולת להגדיר פרמטר Connection שמתייחס לספק של Jet 4. בעת הפניה אל מסד נתונים שנמצא בקובץ אחר, ייתכן שתרצה לכלול פרמטר Data Source שמצביע על המיקום הפיסי של מסד הנתונים שאינו נמצא בפרויקט הנוכחי.

הקוד שלפניך הוא דוגמה פשוטה לפתיחת מסד הנתונים המוכר Northwind. שים לב כי משפט Dim מצהיר ויוצר הפניה אל cnnNorthwind בתור אובייקט Connection. השימוש בשיטה Open ב-cnnNorthwind הופך את מסד הנתונים לזמין עבור שאר חלקי השיגרה. שים לב כי הפרמטרים Provider (ספק) ו- Data Source (מקור נתונים) מופיעים בין גרשיים. פרמטר Provider מצביע על ספק OLE DB של Jet 4, ופרמטר Data Source מצביע על המיקום הפיסי של מסד הנתונים Northwind.

```
Sub OpenMyDB()
Dim cnnNorthwind As New Connection
Dim rsCustomers As Recordset
' Create the connection.
    cnnNorthwind.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office\Office\" & _
        "Samples\Northwind.mdb;"
' Create recordset reference and set its properties.
    Set rsCustomers = New ADODB.Recordset
    rsCustomers.CursorType = adOpenKeyset
    rsCustomers.LockType = adLockOptimistic
' Open recordset and print a test record.
    rsCustomers.Open "Customers", cnnNorthwind, , , adCmdTable
    Debug.Print rsCustomers.Fields(0).Value, rsCustomers.Fields(1).Value
    rsCustomers.Close
    cnnNorthwind.Close
End Sub
```

לאחר יצירת הפניה אל החיבור, הקוד יוצר אובייקט Recordset. הקוד יוצר הפניה אל משתנה האובייקט המציין את ערכת הרשומות, ולאחר מכן מציב ערכים בכמה מאפיינים של ערכת הרשומות. קטע הקוד האחרון פותח את ערכת הרשומות ומדפיס שני שדות מתוך הרשומה הראשונה. השיטה Open של אובייקט Recordset יכולה להפנות חיבור אל מסד נתונים וגם אל מקור רשומות כלשהו במסד הנתונים. הקוד בוחר את כל הרשומות מהטבלה Customers שבמסד הנתונים Northwind. השיטה Open הופכת מלכתחילה את הרשומה הראשונה לזמינה עבור היישום.

שתי השורות המסיימות את קטע הקוד האחרון, סוגרות את ערכת הרשומות ולאחר מכן גם את החיבור. סגירת חיבור הופכת את כל האובייקטים שמפנים אליו, כגון אובייקט Recordset, לבלתי פעילים. כל ניסיון להגדיר מאפיינים או להפעיל שיטות

עבור ערכת רשומות שמפנה לחיבור סגור, יוצר שגיאת זמן ריצה. יצירת חיבור בצורה משתמעת עשויה להיות בחירה טובה יותר, מכיון שהאובייקט יכול להשתמש בחיבור רק במשך קיום האובייקט.

הקוד שלפניך פותח גם הוא ערכת רשומות המבוססת על הטבלה Customers במסד הנתונים Northwind ומדפיס את הרשומה הראשונה. אך הפעולה מבוצעת באמצעות קוד פשוט יותר ומספר שורות קוד קטן יותר, כיון שהחיבור נותר בצורה משתמעת תוך התבססות על מספר הגדרות ברירת מחדל גדול יותר.

```
Sub OpenFast()
Dim rsCustomers As Recordset
    Set rsCustomers = New ADODB.Recordset
' Less code, but potentially greater resource consumption
    rsCustomers.Open "customers", "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office\Office\" & _
        "Samples\Northwind.mdb;"
    Debug.Print rsCustomers.Fields(0), rsCustomers.Fields(1)
    rsCustomers.Close
End Sub
```

מאחר שלא קיים חיבור מפורש, השיגרה OpenFast אינה צריכה להצהיר על אובייקט חיבור (ולכן אינה חייבת לפתוח או לסגור אותו). כפי שניתן לראות, השיטה Open של אובייקט ערכת רשומות יכולה להכיל את נתוני החיבור החיוניים של ספק ושל מקור נתונים. הקוד שהוצג מכיל רק פרמטר אחד נוסף – מקור ערכת הרשומות הוא הטבלה Customers. השיטה Open מתבססת על הגדרות ברירת המחדל CursorType ו-LockType, שמייצגות את המאפיינים קדימה בלבד (forward-only) וקריאה בלבד (read-only). הגדרות אלו נועדו לפעולות מהירות מאוד, אך הן אינן מספקות תפקודיות רבה. אך אם ההגדרות הולמות את צרכיך ומאפשרות להפנות את תשומת ליבך להיבטים אחרים של פיתוח יישומים, ייתכן שהן הבחירה הטובה ביותר.

המאפיין Mode (מצב). כברירת מחדל, השיטה Open של האובייקט Connection יוצרת מסד נתונים עבור גישה משותפת. לעומת זאת, באפשרותך להגדיר את המאפיין Mode של אובייקט זה בתור אחת משבע ההגדרות שמעניקות רמות שונות של גישה מוגבלת למסד נתונים. הגדרות מצב אלו מתייחסות לכל ערכות הרשומות והפקודות שמקצות חיבור למאפיין שלהן ActiveConnection. הקוד שלפניך מציג את השפעת הגדרת המצב קריאה-בלבד על היכולת לעדכן ערכת רשומות.

```
Sub OpenLookOnly()
Dim cnn1 As New Connection
Dim rsCustomers As Recordset
    'cnn1.Mode = adModeRead
    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office\Office\" & _
        "Samples\Northwind.mdb;"
```

```
'Use the next line to determine the default Mode setting.
    Debug.Print cnn1.Mode
    Set rsCustomers = New ADODB.Recordset
    rsCustomers.ActiveConnection = cnn1
    rsCustomers.Open "Customers"

' An adModeRead setting for cnn1.Mode causes an error in this procedure.
' Remove the comment from the cnn1.Mode line to see an error here.
    rsCustomers.Fields("CustomerID") = "xxxxx"
    rsCustomers.Update
    Debug.Print rsCustomers.Fields("CustomerID")
    rsCustomers.Close
End Sub
```

השיגרה OpenLockOnly מצהירה על אובייקט Connection חדש בשורת הקוד הראשונה שלה. השורה השלישית (אם תבטל את סימונה בתור הערה) מגדירה את המאפיין Mode של החיבור בתור adModeRead כדי לאפשר גישה מסוג קריאה-בלבד. אם נעיין בשתי שורות הקוד הבאות, נראה כי השיטה Open הופכת את ערכת הרשומות rsCustomer לזמינה. זוג השורות הבא מנסה לעדכן את ערך השדה CustomerID ברשומה הראשונה. אם תסיר את ההערה בשורה השלישית, עדכונים אלה יגרמו לשגיאה, מכיון שלא ניתן לעדכן מסד נתונים לקריאה-בלבד.

הטבלה שלפניך מתארת את שמונה הקבועים שבאמצעותם ניתן להגדיר את המאפיין Mode של חיבור. באפשרותך לנצל קבועים אלה כדי לשלוט בסוג העריכה שיכולים לבצע משתמש אחד או יותר באמצעות חיבור לערכת רשומות.

קבועים שמגדירים את המאפיין Mode של אובייקט החיבור

קבוע	ערך	תפקיד
adModeUnknown	0	הרשאות לא הוגדרו או נקבעו
adModeRead	1	הרשאת קריאה-בלבד
adModeWrite	2	הרשאת כתיבה-בלבד
adModeReadWrite	3	הרשאת קריאה/כתיבה
adModeShareDenyRead	4	מונע מאחרים לפתוח מקור רשומות עם זכויות קריאה
adModeShareDenyWrite	8	מונע מאחרים לפתוח מקור רשומות עם זכויות כתיבה
adModeShareExclusive	12	מונע מאחרים לפתוח את החיבור
adModeShareDenyNone	16	גישה משותפת (ברירת המחדל)

השיטה OpenSchema. השיטה OpenSchema של האובייקט Connection מאפשרת ליישום לעיין באובייקטים שבאוספים הזמינים באמצעות חיבור מבלי למספר את האלמנטים ברשימה. פלט השיטה עשוי להכיל מידע על טבלאות, תצוגות, שגרות,

אינדקסים ועוד. הפרטים הספציפיים תלויים באופן בו ספק OLE DB נתון מיישם את יכולות השיטה הכלליות. הקוד שלפניך מנצל את השיטה OpenSchema יחד עם ספק Jet 4 כדי להציג את התצוגות הזמינות באמצעות חיבור.

```
Public Sub OpenSchemaX()
Dim cnn1 As New ADODB.Connection
Dim rstSchema As ADODB.Recordset

    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office\Office\" & _
        "Samples\Northwind.mdb;"

    Set rstSchema = cnn1.OpenSchema(adSchemaTables)

' Print just views; other selection criteria include
' TABLE, ACCESS TABLE, and SYSTEM TABLE.
    Do Until rstSchema.EOF
        If rstSchema.Fields("TABLE_TYPE") = "VIEW" Then
            Debug.Print "View name: " & rstSchema.Fields("TABLE_NAME") & vbCr
        End If
        rstSchema.MoveNext
    Loop
    rstSchema.Close
    cnn1.Close
End Sub
```

השיגרה פותחת בהצהרה על חיבור ועל ערכת רשומות. ערכת הרשומות מכילה את הפלט שיצרה השיטה OpenSchema. ארגומנט השיטה OpenSchema מציין כי אלמנטים של התחום Tables של סכימת מסד הנתונים יציבו ערכים ברשומות. לעומת זאת, השיטה OpenSchema עוקבת אחר סוגי טבלה אחדים, כולל תצוגות, טבלאות משתמש רגילות, טבלאות מערכת מיוחדות, טבלה נוספת של אובייקטי Access וטבלאות מקושרות. הקוד שהוצג מדפיס את פלט השיטה עבור תצוגות בלבד.

האובייקט Recordset

ערכת רשומות היא מבנה תוכניתי (programmatic) לטיפול ברשומות. ניתן לבסס את הרשומות על טבלה או על תצוגה בפרויקט הנוכחי, או על קובץ אחר, משפט SQL או פקודה שמחזירה שורות. אפשרויות הטיפול בערכת רשומות תלויות בספק OLE DB שלה ובתכונות מקור הנתונים המקוריים.

באפשרותך לשלוף ערכות רשומות באמצעות אוביקטים אחרים, כגון חיבורים ופקודות, אך מיגוון המאפיינים והשיטות העשיר של האובייקט Recordset הופך אותו לבחירה הטבעית לביצוע חלק גדול מעיבוד הנתונים. ניתן להשתמש בערכות רשומות כדי לבצע פעולות רבות כנגד קבוצת שורות: ניווט בין שורות; הדפסת תוכן השורות,

כולן או מקצתן; הוספה, עדכון ומחיקה של רשומות; חיפוש רשומות; וסינון רשומות כדי לבחור קבוצת-משנה כלשהי מתוך ערכת הרשומות המלאה. מבחינה היסטורית, ערכות רשומות היו ונותרו אובייקטים לא מתמידים – הן קיימות רק כל עוד הן פתוחות בתוכנית. גירסה 2.1 של ADO המשווקת יחד עם Access 2000 כוללת ערכות רשומות מתמידות, אותן אפשר לשמור בדיסק ולפתוח במועד מאוחר יותר.

המאפיין ActiveConnection. מאפיין ערכת רשומות זה מאפשר ליישום לנצל חיבור פתוח כדי לתמוך בערכת רשומות. ניתן לקבוע מאפיין זה בזמן כלשהו לאחר הגדרת האובייקט עבור ערכת הרשומות. השימוש במאפיין זה מפשט את משפט השיטה Open של ערכת הרשומות, כיון שהוא מבטל את הצורך לכלול את נתוני החיבור. בעת הגדרת המאפיין מראש, אינך צריך אפילו להתייחס לחיבור קיים במשפט השיטה Open.

השיטה Open. שיטת ערכת רשומות זו מהווה נתיב מקובל להפיכת ערכת רשומות לזמינה בשיגרה. ארגומנט המקור הוא הארגומנט הקריטי ביותר עבורה. הוא מציין את מקור הנתונים שעליו מבססת השיטה את האובייקט שהיא פותחת. אפשרויות טיפוסיות של ארגומנט המקור כוללות טבלה, משפט SQL, קובץ ערכת רשומות שמור או שיגרה מאוחסנת. משתמשים בארגומנט Options של השיטה Open כדי לתאר את סוג המקור בעת פתיחת ערכת רשומות.

סוג הסמן. סוג הסמן הוא אחד הרכיבים הבסיסיים ביותר של ערכת רשומות. סוג הסמן קובע את אופן הניווט בערכת הרשומות ואת סוגי הנעילות שניתן להחיל עליה. ADO תומכת בארבעה סוגי סמן:

- **Dynamic**. סוג סמן זה מאפשר למשתמשים להציג שינויים במקור נתונים, שביצעו משתמשים אחרים. הוא מאפשר פונקציות אחזקה של ערכת רשומות, כגון הוספה, שינוי ומחיקה של רשומות, ומרשה ניווט דו-כיווני במסד נתונים, מבלי להסתמך על סימניות.
- **Keyset**. סמן זה כולל את רוב המאפיינים של סמן דינמי, למעט גישה מיידית לשינויים שביצעו משתמשים אחרים במקור נתונים. דרך אפשרית להצגת שינויים כאלה היא על ידי הפעלת השיטה Requery של ערכת רשומות.
- **Static**. סמן זה הוא תמונה של ערכת רשומות בנקודת זמן מסוימת. הוא מאפשר ניווט דו-כיווני. לא ניתן להציג שינויים שביצעו משתמשים אחרים במסד הנתונים. גירסה 4 ואילך של Microsoft Internet Explorer תומכות בסוג זה בהיותו סמן צד-הלקוח הגמיש ביותר של הדפדפן.
- **Forward-only**. סמן זה, הנקרא לפעמים סמן ברז-שריפה (hydrant), פועל בכיוון אחד ומסוגל להאיץ את ביצועי הסמן. זהו סמן ברירת המחדל של ADO. אם אתה זקוק לסוג סמן אחר, עליך להגדיר את המאפיין CursorType, בטרם תפתח את ערכת הרשומות.



הערה:
הגדרת סוג הסמן מתקשרת עם הגדרות סוג הנעילה. אם תגדיר סמן מסוג forward-only בעל סוג נעילה שונה מ- read-only (adLockReadOnly), ADO ידרוס את הגדרת סוג הסמן שקבעת. לדוגמה, אם קובעים נעילה אופטימית, ADO ממיר אוטומטית סמני forward-only לסוג ketset.

המאפיין LockType. מאפיין זה מתקשר חלקית עם סוג הסמן, מכיון שהוא קובע כיצד יטפלו משתמשים בערכת רשומות. הגדרה אחת של סוג נעילה (adLockReadOnly) תואמת באופן ייחודי לסמנים מסוג forward-only. זהו סוג ברירת המחדל לנעילה. הטבלה שלפניך מתארת את ארבע ההגדרות האפשריות של המאפיין LockType. ההגדרה adLockBatchOptimistic מיועדת במיוחד עבור מסדי נתונים מרוחקים, כגון SQL Server או Oracle, בניגוד למסד נתונים מקומי Jet. פרק 12 מרחיב בנושא זה.

קבועים שמגדירים את המאפיין LockType של אובייקט החיבור

קבוע	ערך	תפקיד
adLockReadOnly	1	גישה לקריאה-בלבד (ברירת המחדל)
adLockPessimistic	2	נועל רשומה מייד לאחר שהמשתמש מתחיל בעריכתה
adLockOptimistic	3	נועל רשומה רק לאחר שהמשתמש מחיל את השינויים על מסד הנתונים
adLockBatchOptimistic	4	מאפשר פעולות עריכה באצוות רשומות טרם הניסיון לעדכן מסד נתונים מרוחק מתוך אצוות רשומות מקומית



הערה:
תוכל לקבוע אם ערכת הרשומות בה אתה מטפל תכלול רכיבים תפקודיים מסוג מסוים, על ידי שימוש בשיטה Supports. כל שעליך לעשות הוא להציב בסוגריים את הקבוע שמייצג את הרכיב התפקודי הרצוי, בעת הפעלת השיטה. ערך ההחזרה True מציין כי ערכת הרשומות כוללת את הרכיב התפקודי הנדון. התיעוד המקוון של Supports מתאר את שמות הקבועים. חפש את CursorOptionEnum בעזרת סורק האובייקטים (Object Browser), כדי להציג רשימת קבועים עבורם Supports מחזירה True או False.

ניווט בערכת רשומות. ארבע שיטות מאפשרות לנווט בערכת רשומות על ידי שינוי מיקום הרשומה הנוכחית:

- **MoveFirst**. שיטה זו הופכת את הרשומה הראשונה בערכת הרשומות לרשומה הנוכחית. סדר הרשומות תלוי באינדקס הנוכחי, ואם לא קיים אינדקס, הסדר תלוי בסדר ההכנסה. השיטה פועלת בכל סוגי הסמנים. השימוש בשיטה עם סמני forward-only עשוי לאלץ ביצוע חוזר של הפקודה שיצרה את ערכת הרשומות.

- **MoveLast**. שיטה זו קובעת את הרשומה האחרונה בערכת רשומות כרשומה הנוכחית. היא דורשת סוג סמן התומך בתנועה לאחור, או לפחות כזו שמבוססת על סימניות. השימוש בשיטה עם סמן forward-only יוצר שגיאת זמן ריצה.
- **MoveNext**. שיטה זו ממקמת מחדש את הרשומה הנוכחית מקום אחד קדימה ביחס למיקומה הנוכחי (בכיוון הרשומה האחרונה שבערכת הרשומות). אם הרשומה הנוכחית היא גם הרשומה האחרונה, המאפיין EOF של ערכת הרשומות יקבל את הערך True. אם השיטה נקראת בזמן שערך המאפיין EOF הוא True, תתקבל שגיאת זמן ריצה.
- **MovePrevious**. שיטה זו מזיזה את מיקום הרשומה הנוכחית רשומה אחת לאחור. אם הרשומה הנוכחית היא הראשונה, המאפיין BOF של ערכת הרשומות יקבל את הערך True. אם השיטה נקראת בזמן שערך המאפיין BOF הוא True, תתקבל שגיאת זמן ריצה. השיטה יוצרת שגיאת זמן ריצה גם במקרה שמפעילים אותה עם סמן forward-only.

השיטה Move (מעבר אל) פועלת בצורה שונה מארבע שיטות הניווט בערכת הרשומות, מכיון שהיא יכולה להעביר את מיקום הרשומה הנוכחית מספר רשומות משתנה בשני הכיוונים. לציון תנועה בכיוון הרשומה האחרונה משתמשים בארגומנט חיובי, ואילו לציון תנועה בכיוון הרשומה הראשונה משתמשים בארגומנט שלילי. אם התנועה תעבור את הרשומה הראשונה או האחרונה, Move תיתן למאפיין BOF או EOF את הערך True. אם ערך המאפיין הוא True, השיטה תיצור שגיאת זמן ריצה. התנועה היא יחסית לרשומה הנוכחית, אלא אם מציינים פרמטר Start המאפשר לתנועה להתחיל ברשומה הראשונה או האחרונה.

ניתן לשפר את ביצועי השיטה Move בכמה דרכים, על ידי הגדרת המאפיין CacheSize של ערכת הרשומות לערך גדול מ-1 שהוא ערך ברירת המחדל. שינוי הערך של CacheSize גורם ל-ADO לאחסן מספר רשומות קבוע בזיכרון המקומי של תחנת העבודה. אחזור רשומות מהזיכרון מהיר בהרבה מאחזורן מאמצעי אחסון של הספק, ולכן ניתן להאיץ את הניווט ברשומות באמצעות Move על ידי הגדלת ערך CacheSize. סמן מסוג forward-only ו-CacheSize מוגדל, מאפשרים גלילה קדימה ואפילו אחורה. אם הגדרת המטמון זהה למספר הרשומות שבערכת הרשומות, תוכל לגלול את ערכת הרשומות למלוא גודלה בשני הכיוונים. המאפיין CacheSize אינו מאפשר גלילה לאחור באמצעות השיטה MovePrevious (לשם כך משתמשים בשיטה Move עם ארגומנט שלילי).

השיטה Find. שיטה זו מחפשת אחר הרשומה הראשונה העונה על קריטריון בחירה מוגדר. השיטה דומה להפליא לאוסף השיטות Find של גרסאות מוקדמות של Access, אך הגירסה החדשה מאופיינת בתחביר ובתפקוד שונים. במקום לנסות לפרט את קווי הדמיון והשוני בין הגרסאות, מוטב ללמוד את התחביר ואופן הפעולה של הגירסה החדשה.

הגירסה החדשה של השיטה Find קולטת עד ארבעה ארגומנטים. הארגומנט הראשון, קריטריון החיפוש, הוא ארגומנט נדרש. התחביר דומה לזה של פסוקיות WHERE במשפט SQL. אם אין מציינים ארגומנט נוסף כלשהו, השיטה עורכת חיפוש אחר רשומה העונה על הקריטריון החל ברשומה הנוכחית ועד הרשומה האחרונה. לאחר שנמצאה רשומה תואמת, עליך לעבור על פניה, כדי למצוא רשומה תואמת נוספת. אם לא נמצאה רשומה כזו, השיטה נותנת למאפיין EOF את הערך True. עיין בעזרה המקוונת לקבלת תיאור של שלושת הארגומנטים האופציונליים הנותרים.

המאפיין Sort. מאפיין זה יכול להשפיע על תוצאות פעולתן של השיטות Find ו-Move. מאפיין זה מייעד שדה אחד או יותר לקביעת סדר הצגת השורות. ערך המאפיין Sort מאפשר לקבוע סדר מיון עולה או יורד בהתאם לשדה כלשהו. ברירת המחדל היא סדר מיון עולה. ערך המאפיין אינו משפיע בפועל על סדר השורות, אלא על סדר הזמינות שלהן בערכת הרשומות.

המאפיין Filtered. מאפיין זה של ערכת הרשומות מגדיר ערכת רשומות חדשה שהיא גירסה מסוננת של ערכת הרשומות המקורית. למאפיין זה יש יישומים מיוחדים עבור סינכרון מסד נתונים ועדכון באצווה של מקור נתונים מרוחק, אך הוא יכול לשמש חלופה פשוטה להגדרת ערכת רשומות חדשה שמבוססת על משפט SQL. אם אתה זקוק רק לקבוצת-משנה למטרה כלשהי, מאפיין זה יענה היטב על צרכיך.

השיטה AddNew. שיטה זו מוסיפה רשומות חדשות לערכת רשומות. לאחר הפעלת השיטה, מגדירים את ערכי השדות בשורה החדשה. אחר כך עוזבים את הרשומה באמצעות השיטה Move, או קוראים לשיטה Update מבלי לצאת מהשורה החדשה (תוכל לשנות את הערכים בשדה באמצעות שתי טכניקות דומות: עדכון שדות על ידי מתן ערכים חדשים ואחר כך עזיבת הרשומה. לחילופין, קריאה לשיטה Update. אפשר למחוק רשומה על ידי ניווט אליה וקריאה לשיטה Delete. הרשומה שנמחקה נשארת הרשומה הנוכחית עד המעבר לאחרת).

הדפסת ערכי שדה. השיגרה הפשוטה שלפניך פותחת מקור נתונים ולאחר מכן מדפיסה את שורות מסד הנתונים בצורה סדרתית. השיגרה כוללת לולאה שעוברת על כל הרשומות ומדפיסה את שני השדות הראשונים של כל רשומה.

```
Sub EasyLoop()

Dim rsCustomers As Recordset
Dim cnn1 As ADODB.Connection

    Set rsCustomers = New ADODB.Recordset
    rsCustomers.ActiveConnection = CurrentProject.Connection
    rsCustomers.CursorType = adOpenKeyset
    rsCustomers.LockType = adLockOptimistic
    rsCustomers.Open "MyTable", , , , adCmdTable
```

```
    'rsCustomers.Open "customers", "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office\Office\" & _
        "Samples\Northwind.mdb;"

    'rsCustomers.Sort = "customerid"

'Loop through recordset.
    Do Until rsCustomers.EOF
        Debug.Print rsCustomers.Fields(0), rsCustomers.Fields(1)
        rsCustomers.MoveNext
    Loop

    rsCustomers.Close
End Sub
```

אחד החסרונות של השיגרה EasyLoop הוא שהיא מדפיסה רק ערכי שדות מבוקשים. השיגרה EasyLoop2 שלהלן, מתגברת על הקושי. אין זה משנה מה מספר השדות במקור הנתונים של ערכת הרשומות, השיגרה מדפיסה אותם במלואם ללא התערבות.

```
Sub EasyLoop2()

Dim rsCustomers As Recordset
Dim fldMyField As Field
Dim strForRow As String

    Set rsCustomers = New ADODB.Recordset
    rsCustomers.Open "customers", & _
        "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office\Office\" & _
        "Samples\Northwind.mdb;"

' Loop through recordset and fields with rows.
    Do Until rsCustomers.EOF
        strForRow = ""
        For Each fldMyField In rsCustomers.Fields
            strForRow = strForRow & fldMyField & "; "
        Next fldMyField
        Debug.Print strForRow
        rsCustomers.MoveNext
    Loop

    rsCustomers.Close
End Sub
```

קבוצת השורות הראשונה והקבוצה האחרונה בשתי השגרות, פותחות ערכת רשומות וסוגרות אותה. השיגרה EasyLoop2 מקננת לולאה מסוג For בתוך לולאה מסוג Do. הלולאה הפנימית For ממספרת את שדות השורה ויוצרת בכל שורה מחרוזת אחת שמורכבת מכל ערכי השדות (בתחילת הלולאה מנקים את המחרוזת מכל ערכיה כדי לבצע את התהליך פעם נוספת עבור השורה הבאה).

לולאה היא דרך קלה לביצוע פעולה על שורות ועמודות של ערכת רשומות. לעומת זאת, אין זו הדרך היעילה ביותר לאחזור ערכי שדה של ערכת רשומות. השיגרה EasyLoop שלפניך מנצלת את השיטה GetString כדי לאחזר ולהדפיס את כל השדות בכל שורות ערכת הרשומות, וזאת בשלב אחד בלבד. השיטה GetString מחזירה ערכת רשומות בתור מחרוזת. היא יכולה לקבל עד חמישה ארגומנטים; הקוד שלפניך מנצל שלושה מהם. מציבים את הקבוע adClipString בתור ארגומנט ראשון – זוהי הבחירה היחידה. הארגומנט מציין את תבנית הייצוג של ערכת הרשומות בתור מחרוזת. הארגומנט השני מציין את מספר שורות ערכת הרשומות שיש להחזיר. הקוד מחזיר חמש שורות. אם אין מגדירים ערך כלשהו לארגומנט, השיטה תחזיר את כל השורות של ערכת הרשומות.

הארגומנט השלישי מציב נקודה-ופסיק (;) בתור תו מפריד בין עמודות בשורה. מפריד ברירת המחדל הוא טאב. הארגומנטים הרביעי והחמישי, שאינם מופיעים בדוגמה, מציינים מפריד עמודות וביטוי לייצוג ערכי null. ערכי ברירת המחדל של ארגומנטים אלה הם תו החזרת גררה (CR) ומחרוזת באורך אפס.

```
Sub NoEasyLoop()
Dim rsCustomers As Recordset
    Set rsCustomers = New ADODB.Recordset
    rsCustomers.Open "customers", _
        "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office\Office\" & _
        "Samples\Northwind.mdb;"
'Print records without a loop.
    Debug.Print rsCustomers.GetString(adClipString, 5, "; ")
    rsCustomers.Close
End Sub
```

השיטה GetString מחליפה שתי לולאות מקוננות. אם ברירות המחדל מתאימות לצרכיך, תוכל להשתמש בשיטה ללא ארגומנטים כלשהם. הדבר מאפשר לשלוף ערכים מתוך ערכת רשומות. למרות שלולאות מקוננות הן דרך אינטואיטיבית לשלוף ערכים מתוך ערכת רשומות, השיטה GetString יכולה להביא לתוצאה דומה בשורת קוד אחת בלבד.

הוספת רשומה. הקוד הבא מבצע משימה חדשה: הוספת רשומה חדשה למקור נתונים.

```
Sub AddARecord()

Dim rsMyTable As Recordset

' Set your cursor so that it is not read-only to delete.
   Set rsMyTable = New ADODB.Recordset
   rsMyTable.ActiveConnection = CurrentProject.Connection
   rsMyTable.Open "MyTable", , adOpenKeyset, adLockOptimistic, adCmdTable

' Invoke the AddNew method.
   rsMyTable.AddNew
     rsMyTable.Fields("Column1").Value = 16
     rsMyTable.Fields("Column2").Value = 17
     rsMyTable.Fields("Column3").Value = 18
   rsMyTable.Update

End Sub
```

השגרות EasyLoop, EasyLoop2 ו-NoEasyLoop מקבלות כולן את הגדרות ברירת המחדל של סוג הסמן וסוג הנעילה, אך השיגרה AddARecord אינה נוהגת כך. כזכור, ברירות המחדל הן סמן מסוג forward-only ונעילה מסוג read-only. הגדרות אלו מתאימות רק עבור הדפסת תוכן ערכת רשומות. אך יש צורך בסוג סמן ובסוג נעילה המאפשרים עדכון ערכת רשומות, כשהמשימה הנדרשת כוללת הוספה, עריכה או מחיקה של רשומות. הארגומנטים adOpenKeyset ו-adLockOptimistic של השיטה Open מאפשרים להוסיף שורות חדשות לערכת רשומות. שים לב שההגדרה ActiveConnection בקוד לעיל אינה מפנה לפרויקט הדוגמה Northwind, אלא מפנה לחיבור הפרויקט הנוכחי. כשעליך להתייחס למקור נתונים בפרויקט הנוכחי, פעל לפי כללי התחביר. משפט החיבור מציב מפורשות טבלה בפרויקט הנוכחי כמקור הנתונים של ערכת הרשומות. קיימים מקורות חלופיים אחדים, כולל טקסט של משפט SQL, שיגרה מאוחסנת, קובץ חיצוני שנשמר בתבנית מיוחדת ועוד.

כדי להוסיף רשומה באמצעות השיטה AddNew, קוראים לשיטה, מפעילים משפטי הצבה כדי למלא את הרשומה החדשה בערכים, ולאחר מכן מפעילים את השיטה Update. אין חובה מוחלטת לקרוא ל-Update; ניתן פשוט לעזוב את הרשומה הנוכחית החדשה. לדוגמה, תוכל להפעיל את MoveFirst או שיטה אחרת, כדי לנווט לרשומה חדשה.

עריכה או מחיקה של רשומה. הקוד הבא עורך או מוחק רשומה. הקוד אינו משתמש בשיטות Edit או Update כדי לשמור את הרשומות לאחר עריכתן, אלא עוזב את הרשומה. אם אין זה מעשי לעזוב את הרשומה, או אם היישום צריך לאמת את השינויים טרם עזיבת הרשומה, השתמש בשיטה Update.

```
Sub DeleteOrUpdateARecord()
Dim rsMyTable As Recordset

'Set your cursor so that it is not read-only to delete.
    Set rsMyTable = New ADODB.Recordset
    rsMyTable.ActiveConnection = CurrentProject.Connection
    rsMyTable.Open "MyTable", , adOpenKeyset, adLockOptimistic, adCmdTable
' Loop through recordset.
    Do Until rsMyTable.EOF
        If rsMyTable.Fields("Column1") = 16 Then
'            rsMyTable.Fields("Column1") = 88
            rsMyTable.Delete
        End If
        rsMyTable.MoveNext
    Loop

    rsMyTable.Close
End Sub
```

לולאה כדוגמת זו שבשיגרה DeleteOrUpdateARecord יכולה לסייע לך לבחור רשומות למחיקה או לעריכה. השיגרה בודקת כל ערך של השדה Column1 בערכת רשומות בחפשה אחר שדה שערכו 16. כשנמצא השדה המבוקש, השיטה מוחקת את השורה. שים לב שהלולאה כוללת שורת הערה. כדי לעבור משגרת מחיקה לשגרת עדכון, העבר את סימן ההערה משורת ההערה לשורת השיטה Delete.

חיפוש רשומות. שימוש נפוץ נוסף בערכת רשומות, הוא חיפוש רשומה אחת או יותר העונות לקריטריונים מוגדרים. Access 2000 מאפשר לבצע משימה זו בכמה דרכים. בגרסאות מוקדמות של Access, הרבו להשתמש בווריאציה אחת או יותר של השיטה Find. כפי שכבר הזכרנו, Access 2000 כולל גירסה יחידה של השיטה Find השונה במידת-מה מגרסאותיה הקודמות. גם בגירסה זו תוכל להשתמש ברכיבים התפקודיים הדומים של הגירסה החדשה של Find.

הקוד שלפניך מציג יישום פשוט של השיגרה Find, אשר מחפש רשומה שמזהה הלקוח שלה (CustomerID) מתחיל באות D. כשהרשומה מאותרת, השיטה קובעת אותה כרשומה הנוכחית. הקוד מדפיס את השדות CustomerID ו-ContactName כדי שאפשר יהיה לאשר במדויק את הרשומה העונה על קריטריון החיפוש.

```
Sub FindAMatch()
Dim rsCustomers As Recordset
    Set rsCustomers = New ADODB.Recordset
    rsCustomers.ActiveConnection = _
        "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office\Office\" & _
        "Samples\Northwind.mdb;"
```

```
    rsCustomers.Open "Customers", , adOpenKeyset, adLockPessimistic, _
    adCmdTable
    rsCustomers.Find ("CustomerID Like 'D*'")
    Debug.Print rsCustomers.Fields("CustomerID"), _
    rsCustomers.Fields("ContactName")
End Sub
```

אחד החסרונות של הגישה שתוארה הוא בכך שהיא מחפשת מופע תאימות בודד ותהליך החיפוש נעצר מייד לאחר שנמצאה התאמה כזו. הקוד שלפניך מגלה את כל הרשומות התואמות למשפט הקריטריון. יישום פשוט זה מציג היבטים נוספים של גמישות השיטה Find.

```
Sub FindAMatch2()
Dim rsCustomers As Recordset
    Set rsCustomers = New ADODB.Recordset
    rsCustomers.ActiveConnection = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office\Office\" & _
        "Samples\Northwind.mdb;"
    rsCustomers.Open "Customers", , adOpenKeyset, adLockPessimistic,
adCmdTable
    Do
        rsCustomers.Find ("CustomerID Like 'D*'")
        If rsCustomers.EOF Then
            Exit Sub
        End If
        Debug.Print rsCustomers.Fields("CustomerID")
        rsCustomers.MoveNext
    Loop
End Sub
```

הפתרון לאיתור כל הרשומות העונות על קריטריון החיפוש הוא לשבץ את השיטה Find בלולאה מסוג Do. כשהשיטה Find מגדירה את המאפיין EOF של ערכת הרשומות בתור True, לא קיימות רשומות תואמות נוספות. במקרה זה, הקוד מבצע את המשפט Exit Sub ועל ידי כך מסיים את השיגרה. כל עוד Find תמשיך לגלות מופעי תאימות נוספים, השיגרה תדפיס את מזהה הלקוח בחלון Immediate (מיידי). לאחר הדפסת רשומה תואמת, השיגרה מתקדמת מהרשומה הנוכחית אל הבאה אחריה. לולא עשתה כן, השיטה Find היתה חוזרת שוב ושוב לאותה רשומה.

השיטה Find עוברת על ערכת רשומות בצורה סדרתית ובכל פעם מגלה התאמה אחת. השיטה אינה יוצרת גירסה נוספת של ערכת רשומות שכוללת את כל הרשומות העונות על הקריטריון. אם יש צורך בערכת רשומות חדשה או חלופית המכילה את הרשומות התואמות בלבד, עליך לנקוט גישה שונה מזו. המאפיין Filter (מסנן) של ערכת הרשומות עשוי להיות התשובה. מאפיין זה מאפשר להגדיר קריטריון פשוט לשדה,

ומחזיר גירסה מסוננת של ערכת הרשומות המקורית הכוללת רק את הרשומות העונות על הקריטריון שצוין. על ידי הגדרת המאפיין Filter כקבוע כלשהו מתוך רשימת קבועים, תוכל להשיג תוצאות מיוחדות בתהליכי שכפול מסד נתונים, או עדכון מקור נתונים מרוחק. קבוע המסנן adFilterNone מסיר את הגדרת המסנן מתוך ערכת הרשומות ומשחזר את ערכיה המקוריים.

סינון רשומות. שתי השגרות שלפניך מסננות ערכת רשומות המבוססת על הטבלה Customers במסד הנתונים Northwind. השיגרה FilterRecordset מנהלת את השימוש הכולל במאפיין Filter, מדפיסה את קבוצת התוצאות, מנקה את המסנן ולאחר מכן מדפיסה את קבוצת התוצאות פעם נוספת. השיגרה FilterRecordset מתבססת על הפונקציה FilterLikeField המנהלת את הגדרת המאפיין Filter, בהתאם לפרמטרים שהעבירה אליה השיגרה FilterRecordset.

```
Sub FilterRecordset()
Dim rsCustomers As Recordset
' Create recordset variable.
   Set rsCustomers = New ADODB.Recordset
   rsCustomers.ActiveConnection = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=C:\Program Files\Microsoft Office\Office\" & _
      "Samples\Northwind.mdb;"
' Open recordset.
   rsCustomers.Open "Customers", , , , adCmdTable
'Filter recordset.
   Set rsCustomers = FilterLikeField(rsCustomers, "CustomerID", "D*")
   Debug.Print rsCustomers.GetString
' Restore recordset.
   rsCustomers.Filter = adFilterNone
   Debug.Print rsCustomers.GetString
   rsCustomers.Close
End Sub

Function FilterLikeField(rstTemp As ADODB.Recordset, _
   strField As String, strFilter As String) As ADODB.Recordset
'Set a filter on the specified Recordset object and then
'open a new Recordset object.
   rstTemp.Filter = strField & " LIKE '" & strFilter & "'"
   Set FilterLikeField = rstTemp
End Function
```

השיגרה FilterRecordset פותחת ביצירה ובפתיחה של ערכת הרשומות rsCustomer. לאחר מכן היא מיישמת מסנן על ידי קריאה לפונקציה FilterLikeField, אשר קולטת שלושה ארגומנטים ומחזירה ערכת רשומות המבוססת עליהם. FilterRecordset מציבה את ערך ההחזרה המסונן ב-rsCustomer ומדפיסה אותה כדי לאמת את התוצאה.

הארגומנטים שנשלחו אל FilterLikeField כוללים את rsCustomer (ערכת הרשומות), שם שדה שלפיו יש לערוך את הסינון (CustomerID), וערך קריטריון סינון (D*). קריטריון הסינון עשוי לכלול ביטוי חוקי עבור האופרטור Like של הפונקציה FilterLikeField. FilterRecordset מעבירה D* כדי לצמצם את החיפוש לרשומות שערך השדה CustomerID שלהן מתחיל באות D. המאפיין Filter אינו מגביל את הסינון לאופרטור Like בלבד. ניתן גם לבצע סינון באמצעות האופרטורים >, <, >=, <=, <> ו-=. אפשר לכלול את האופרטורים הלוגיים And ו-Or בביטויי הקריטריון, כדי לחבר שני ביטויים או יותר בהתבסס על אופרטורים חוקיים נוספים.

מאפיין Filter מגביל את הקריטריון לביטויים בתצורת FieldName-Operator-Value. אך קיימים קבועים של Filter המאפשרים שימושים מיוחדים. FilterRecordset משתמשת במאפיין adFilterNone כדי לשחזר ערכת רשומות על ידי הסרת המסננים.

שימוש ב-SQL ליצירת ערכת רשומות. עובדה אחרונה שעליך לדעת על ערכות רשומות: כיצד ליצור ערכת רשומות בהתבסס על משפטי SQL. לעיתים קרובות משפטי SQL אינם מורכבים יותר מהמשפט "SELECT * FROM TABLENAME", אך באפשרותך לנצל את מלוא הרכיבים התפקודיים של SQL כדי ליצור ערכות רשומות. תוכל גם להשתמש במשפטי SELECT מורכבים ומרובי טבלאות עם שדות מחושבים המשתמשים בצירופים (joins) פנימיים וחיצוניים אשר מגבילים או מארגנים קבוצות החזרה בעזרת הפסוקיות WHERE, GROUP BY ו- ORDER BY. דרך קלה ליצירת ערכת רשומות מותאמת אישית שמבוססת על משפטי SQL, היא השימוש בפסוקיות WHERE. באפשרותך לשלוף רשומות בצורה סלקטיבית מתוך מקור קיים באמצעות ביטויים מורכבים יותר מאשר ניתן להפעיל בעת שימוש במאפיין Filter.

הקוד שלפניך משתמש בשיטה Open ובמשפט SQL. כשמבססים ערכת רשומות על משפט SQL במקום על טבלה קיימת, מעבירים את משפט SQL ומשתמשים בארגומנט האופציונלי adCmdTable במקום בארגומנט adCmdText. זה כל מה שיש לעשות. לאחר מכן ניתן לנצל את ערכת הרשומות כדי לבנות ערכת רשומות פשוטה כלשהי שמבוססת על טבלה מסוימת. משפטי SQL מורכבים יותר אינם משנים את ההצהרה או את אופן הטיפול בערכת רשומות באמצעות ADO.

```
Sub SQLRecordset()
Dim rsCustomers As Recordset
' Create recordset variable.
   Set rsCustomers = New ADODB.Recordset
   rsCustomers.ActiveConnection = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=C:\Program Files\Microsoft Office\Office\" & _
      "Samples\Northwind.mdb;"

' Open the recordset.
   rsCustomers.Open "SELECT * FROM Customers", , adOpenForwardOnly, _
      adLockReadOnly, adCmdText
   Debug.Print rsCustomers.GetString

   rsCustomers.Close
End Sub
```

האובייקט Field

שדה הוא עמודת נתונים שמכילה ערכים מסוג נתונים זהה. בספריה ADODB, האוסף Fields שייך בלעדית לערכות רשומות וחבריו הם האובייקטים Field. אובייקטים אלה כוללים מאפיינים ושיטות המשמשים לאחסון ושליפה של נתונים.

ערכות רשומות מנצלות את המאפיין Value של האובייקט Field כדי להציג את תוכן העמודה של הרשומה הנוכחית. כשמשנים את הרשומה, ערך זה יכול להשתנות כדי לייצג את תוכן הרשומה החדשה. מאפיינים רבים נוספים של Field מכילים metadata – נתונים **אודות** נתוני הרשומה. המאפיין Name הוא נקודת אחיזה (handle) שמאפשרת ליישום להתייחס לשדה מסוים. המאפיין DefinedSize מאפיין את הגודל המירבי של שדה (בתווים, בשדות Text). המאפיין ActualSize הוא הגודל בפועל, בבתים, של ערך באובייקט Field. המאפיין Attributes מכיל מערך רכיבי מידע על שדה. באפשרותו לציין אם ערך שדה הוא בר-עדכון או אם השדה יכול להכיל ערכי Null.



הערה:

המאפיינים DefinedSize ו-ActualSize משתמשים ביחידות מידה שונות עבור שדות Text. DefinedSize הוא מספר התווים המירבי בשדה, ו-ActualSize הוא מספר הבתים שהשדה מכיל בפועל. שדות Text של Jet 4 מייצגים תווים באמצעות שני בתים לכל תו, ולכן הערך ActualSize שלהם עשוי להיות אפילו כפול מהערך DefinedSize. בשדות נומריים ובשדות Text במסדי נתונים המייצגים תו באמצעות בית יחיד (כגון מסד הנתונים Jet 3.51), הבדל זה אינו קיים.

השיטות של Field – GetChunk ו-AppendChunk – מאפשרות לעבד שדות גדולים של טקסט או של נתונים בינאריים בגושים קטנים שקל יותר לטפל בהם בזיכרון המחשב. השיטה GetChunk משמשת להעביר לזיכרון חלק מהשדה. הארגומנט Size מציין את מספר הבתים שיש לאחזר בהפעלה אחת של השיטה GetChunk. בכל הפעלה רציפה ללא הפרעות של השיטה, מתבצעת קריאת נתונים חדשים מהנקודה בה הסתיימה קריאת הנתונים הקודמים. השיטה AppendChunk מאפשרת לבנות מתוך הזיכרון שדה גדול מאוד המכיל טקסט או נתונים בינאריים, בצורת גושים. בדומה לשיטה GetChunk, גם שיטה זו כותבת בשדה נתונים חדשים החל מהמקום בו הסתיימה הכתיבה הקודמת של AppendChunk. כדי להפעיל את השיטות בצורה נכונה, יש לתת לסיבית adFldLong של המאפיין Attribute באובייקט Field את הערך True.

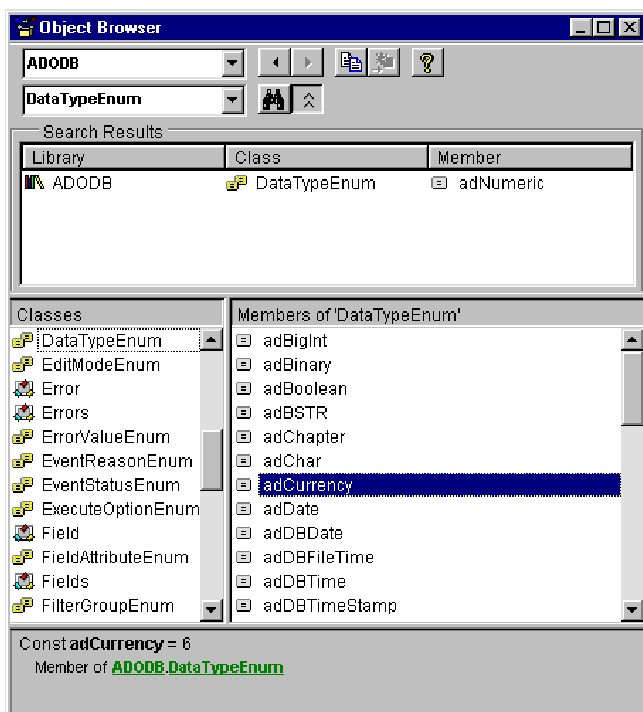
המאפיינים Name ו-Value. השיגרה שלפניך מציגה את השימוש הנפוץ של מאפיינים אלה. השיגרה מציגה את כל שמות השדות לצד ערכיהם. היא יוצרת ערכת רשומות בת רשומה אחת בהתבסס על משפט SQL.

```
Sub FieldNameValue()
Dim cnn1 As ADODB.Connection
Dim rsCustomers As ADODB.Recordset
Dim fldLoop As ADODB.Field
' Open connection and recordset.
   strCnn = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=C:\Program Files\Microsoft Office" & _
      "\Office\Samples\Northwind.mdb;"
   Set cnn1 = New ADODB.Connection
   cnn1.Open strCnn
   Set rsCustomers = New ADODB.Recordset
   rsCustomers.ActiveConnection = cnn1
   rsCustomers.Open "SELECT * FROM Customers " & _
      "WHERE CustomerID='BONAP'", , , , adCmdText
 ' Report field names and values for record.
   For Each fldLoop In rsCustomers.Fields
      Debug.Print fldLoop.Name, fldLoop.Value
   Next fldLoop
End Sub
```

השיגרה מתחילה בפתיחת חיבור ולאחר מכן יוצרת ערכת רשומות בחיבור. משפט SQL שולף את רשומת הלקוח שערך השדה CustomerID שלה שווה ל-BONAP. לולאת Do, הבאה לאחר יצירת ערכת הרשומות, עוברת על שדות ערכת הרשומות. הדפסת המאפיין Name יחד עם המאפיין Value מסייעת להבנת פעולת השיגרה.

המאפיין Type. המאפיין Type של האובייקט Field מציינת את סוג הנתונים שהשדה יכול להכיל. המאפיין מחזיר את אחד מקבועי סוגי הנתונים שבטווח הערכים DataTypeEnum. תוכל להציג את הקבועים האפשריים שבספריה ADODB באמצעות סורק האובייקטים. תרשים 2.5 מציג את הקבועים הללו בחלון **Object Browser**. בחירת סוג שדה מאפשרת לקבוע ערכים חוקיים עבור המאפיין Value של סוג השדה.

הדפסת סוגי נתונים של שדות. שתי השגרות הבאות פועלות יחד כדי לעבד קבועים של סוגי נתונים באמצעות ADO. השיגרה FieldNameType פותחת ערכת רשומות המבוססת על הטבלה Orders במסד הנתונים Northwind. טבלה זו מכילה מיגוון עשיר למדי של סוגי נתונים, ולכן ניתן לנצל אותה כדוגמה לבחינת הנושא של סוגי נתונים. לאחר פתיחת ערכת רשומות, השיגרה עוברת בלולאה על כל השדות שבערכת הרשומות ומדפיסה שם וסוג של כל אובייקט Field. הפונקציה FieldType מתרגמת את הערך המספרי של הקבוע למחרוזת המייצגת את הקבוע. הערך המספרי של הקבוע adCurrency הוא 6. הפונקציה FieldType ממירה את הערך 6 למחרוזת "adCurrency". כעת מדפיסה השיגרה FieldNameType את שמות כל השדות ואת שמות כל הקבועים של סוגי הנתונים.



תרשים 2.5: סורק האובייקטים מציג מבחר קבועים של סוגי נתונים

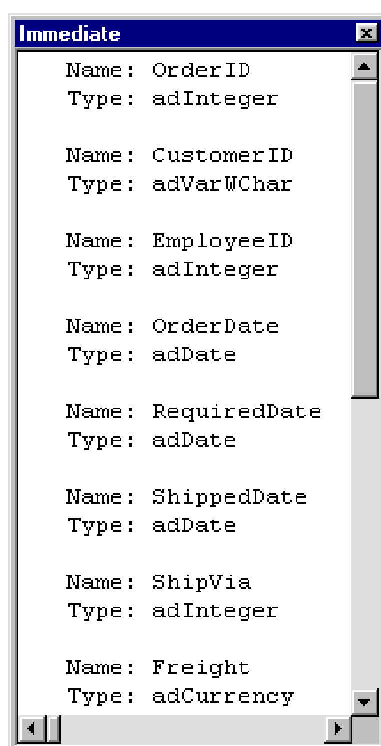
```
Sub FieldNameType()
Dim cnn1 As ADODB.Connection
Dim rsOrders As ADODB.Recordset
Dim fldLoop As ADODB.Field
' Open connection and recordset.
    strCnn = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office" & _
        "\Office\Samples\Northwind.mdb;"
    Set cnn1 = New ADODB.Connection
    cnn1.Open strCnn
    Set rsOrders = New ADODB.Recordset
    rsOrders.ActiveConnection = cnn1
    rsOrders.Open "orders", , , , adCmdTable

' Report field names and types for record.
    For Each fldLoop In rsOrders.Fields
        Debug.Print "  Name: " & fldLoop.Name & vbCr & _
            "  Type: " & FieldType(fldLoop.Type) & vbCr
    Next fldLoop
End Sub
```

```
Public Function FieldType(intType As Integer) As String

    Select Case intType
        Case adVarWChar
            FieldType = "adVarWChar"
        Case adCurrency
            FieldType = "adCurrency"
        Case adInteger
            FieldType = "adInteger"
        Case adDate
            FieldType = "adDate"
    End Select

End Function
```



תרשים 2.6: קטע מתוך הפלט של FieldNameType

תרשים 2.6 מציג קטע מתוך פלט FieldNameType. הקטע כולל לפחות שדה אחד מכל סוג נתונים שממירה הפונקציה FieldType. תוכל להפעיל את FieldNameType ואת FieldType כנגד ערכות רשומות בהתבסס על מקורות נתונים שונים. ייתכן שתפגוש סוג נתונים נוסף לארבעה שברשימה. במקרה זה, שדה Type שבדוחות יהיה ריק. תקן את הבעיה על ידי קביעת ערך השדה. הכנס נקודת עצירה במשפט Debug.Print בלולאת Do

של השיגרה FieldNameType. בדוק את ערך fldloop.Type של שדה שסוגו אינו מוצג, ואחר כך התאם את ערך הקבוע כנגד הקבועים בטווח הערכים DataTypeEnum שמציג סורק האובייקטים (תרשים 2.5). לבסוף, הוסף את המשפט Select Case לשיגרה FieldType כדי להמיר את הקבוע החדש.

חיפוש ערך השדה הגדול ביותר. השיגרה FieldSizes שלפניך מיישמת את המאפיין ActualSize כדי למצוא את הערך הגדול ביותר בשדה CompanyName של הטבלה Shippers במסד הנתונים Northwind. השיגרה מתחילה ביצירת חיבור אל מסד הנתונים ולאחר מכן פותחת ערכת רשומות המבוססת על הטבלה Shippers. חלקה השני של השיגרה מאתר את שם המוביל הארוך ביותר ומציג תיבת הודעה ובה השם ומספר התווים שהוא מכיל.

```
Sub FieldSizes()
Dim cnn1 As ADODB.Connection
Dim rsShippers As ADODB.Recordset
Dim fldLoop As ADODB.Field
Dim intMaxChars As Integer, strMsg As String
Dim strName As String
' Open connection and recordset.
    strCnn = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office" & _
        "\Office\Samples\Northwind.mdb;"
    Set cnn1 = New ADODB.Connection
    cnn1.Open strCnn
    Set rsShippers = New ADODB.Recordset
    rsShippers.ActiveConnection = cnn1
    rsShippers.Open "SELECT * FROM Shippers" , , , , adCmdText
' Find longest shipper's name.
    intMaxChars = 0
    Do Until rsShippers.EOF
        If rsShippers!CompanyName.ActualSize / 2 > intMaxChars Then
            intMaxChars = rsShippers!CompanyName.ActualSize / 2
            strName = rsShippers.Fields("CompanyName")
        End If
        rsShippers.MoveNext
    Loop
    strMsg = "The longest shipper's name is '" & _
        strName  & "' (" & intMaxChars & " characters)."
    MsgBox strMsg, vbInformation, "Programming Microsoft Access 2000"
    rsShippers.Close
End Sub
```

המשתנה intMaxChars עוקב אחר אורך השדה הגדול ביותר. חלקה שני של השיגרה FieldSizes מאתחל את המשתנה ל-0 בטרם יחל ביצוע הלולאה שעוברת על כל הרשומות שבערכת הרשומות. אין הכרח לאפס את intMaxChars, מכיון ש-VBA עושה זאת בעצמו. אך האתחול תורם למעין תיעוד עצמי של השיגרה. כל שם מוביל שמספר התווים שהוא מכיל גדול יותר מהערך הנוכחי של intMaxChars הוא השם הארוך ביותר עד לאותו שלב של הבדיקה. כשהשיגרה מוצאת שם העונה על קריטריון זה, היא מעדכנת את המשתנה intMaxChars ושומרת את השם. שים לב שהשיגרה משתמשת בשתי מערכות של כללים תחביריים כדי להפנות לשדה. משפטים אחדים משתמשים בתו bang (!) שהוא סימון שהיה מקובל בגרסאות ישנות של Access. בסימון הנוכחי, התו bang מפריד בין שם ערכת הרשומות לשם השדה. הדרך החדשה והמקובלת יותר היא ליצור הפניה אל השדה באמצעות האוסף Field. באפשרותך להשתמש באינדקס המספרי, אם הוא ידוע לך, או לתחום את שם השדה בין גרשיים.

האובייקטים Command ו-Parameter

לאובייקטים Command של הספריה ADODB שלושה יתרונות :

- הם מסוגלים לבצע שאילתת בחירה כדי להחזיר קבוצת שורות ממקור נתונים.
- הם מבצעים שאילתת פרמטר, דבר שמאפשר לספק קריטריון חיפוש בזמן ריצה.
- הם תומכים בשאילתות פעולה כנגד מקור נתונים, כדי לבצע פעולות כגון עדכון, מחיקה והוספת רשומות.

כפי שנלמד בסעיפים הבאים, האובייקט **Command** יכול למלא תפקידים נוספים בספריות אחרות.

עליך לייעד אובייקט **Connection** שעליו תופעל פקודה. תוכל ליצור אובייקט Connection בצורה משתמעת בעת ציון פקודה, או לשייך מפורשות אובייקט Connection קיים לפקודה. אפשרויות אלו זהות לאפשרויות ערכות רשומות.

המאפיין CommandTimeout קובע את משך הזמן ש-ADO תמתין לסיום ביצוע פקודה. מאפיין זה קולט ערך מסוג Long שמציין את זמן ההמתנה המירבי בשניות. ערך ברירת המחדל שלו הוא 30. אם פסק הזמן חולף בטרם סיים האובייקט Command את ביצוע הפקודה, ADO תבטל את הפקודה ותחזיר שגיאה. אובייקט Connection תומך אף הוא במאפיין CommandTimeout, אך אינו תלוי במאפיין CommandTimeout של אובייקט Command. מאפיין CommandTimeout של אובייקט Command אינו יורש את הגדרת המאפיין CommandTimeout של אובייקט Connection.

המאפיין CommandType. קיימים למעשה סוגים אחדים ושונים של האובייקט Command. המאפיין CommandType מגדיר את סוג אובייקט Command. תוכל לבסס את הפקודה על משפט SQL, על טבלה או על שיגרה מאוחסנת, כפי שמציגה הטבלה הבאה. הסיבה העיקרית לשימוש בהגדרת המאפיין CommandType היא לאפשר יצירת

אובייקט Command שמבוסס על משפט SQL. החלפת ערך ברירת המחדל של הקבוע CommandType בערך אחר, עשויה להאיץ את ביצוע הפקודה. לכן, אם המקור ידוע לך, מוטב שתגדיר את הקבוע הזה.

המאפיין CommandText. כדי לכתוב משפט SQL שייבצע את הפקודה, משתמשים בהגדרה CommandText של האובייקט Command. באפשרותך גם להגדיר את המאפיין בתור שם של שיגרה מאוחסנת. בעת הפעלת משפט SQL, ניתן להשתמש במאפיין Prepared כדי לציין שהמשפט מיועד להידור (קומפילציה) ולאחסון בשרת מסד הנתונים. הדבר מאט את הביצוע הראשון של הפקודה, אך במקביל מאיץ את ביצועיה בהמשך. תן למאפיין Prepared את הערך True כדי להדר משפט SQL.

קבועי CommandType

קבוע	ערך	תפקיד
adCmdText	1	מאפשר להפעיל פקודה המבוססת על משפט SQL, שיגרה מאוחסנת או אפילו טבלה. בדרך כלל תשמור הגדרה זו עבור משפט SQL
adCmdTable	2	מבסס את קבוצת ערכי ההחזרה על טבלה שעוצבה קודם לכן. מחזיר את כל עמודות הטבלה בהתאם למשפט SQL שנוצר באופן פנימי
adCmdStoredProc	4	מפעיל פקודה המבוססת על טקסט עבור שיגרה מאוחסנת
adCmdUnknown	8	לא קיים מפרט כלשהו לגבי סוג טקסט הפקודה. זו ברירת המחדל
adCmdFile	256	מעריך פקודה בהתאם לשם הקובץ עבור ערכת רשומות תמידית
adCmdTableDirect	512	מעריך פקודה בהתאם לשם הטבלה. מחזיר את כל העמודות בטבלה ללא קוד SQL מתווך כלשהו

השיטה Execute. השיטה Execute של אובייקט Command מפעילה את קוד הרקע של האובייקט Command (שאילתה, משפט SQL או שיגרה מאוחסנת). תוכל לציין עד שלושה ארגומנטים. הארגומנט הראשון מאפשר לאובייקט Command למסור לשיגרה המפעילה אותו את מספר הרשומות שעליהן פעל. הארגומנט השני יכול להיות מערך Variant שמכיל פרמטרים להפעלת הפקודה. הארגומנט השלישי מודיע ל-ADO כיצד להעריך את המקור. ארגומנט זה יכול להיות אחד מתוך רשימת שמות הקבועים המוצגים בטבלה "קבועי CommandType", לעיל.

השיטה CreateParameter. השיטה CreateParameter של אובייקט Command יוצרת פרמטר חדש של פקודה. לאחר יצירת הפרמטר, תוכל לנצל את השיטה Append כדי להוסיף את הפרמטר לאוסף Parameters של פקודה. בטרם תפעיל שאילתת פרמטר, עליך לספק לפרמטר ערך.

יצירת ערכת רשומות באמצעות שאילתת בחירה. אחת המטלות המיידיות שניתן לבצע בעזרת אובייקט Command היא ליצור ערכת רשומות שמבוססת על שאילתת בחירה (select query). האובייקט Command מפעיל את שאילתת הבחירה ומייצג את הערכים המוחזרים שלו. בשלב זה יכול הקוד שכתבת לפתוח אובייקט Recordset המבוסס על הקבוצה המוחזרת מהאובייקט Command. זה בדיוק מה שמבצעת השיגרה SelectCommand שלפניך. לשיגרה שני חלקים: האחד יוצר את האובייקט Command וחיבור שמקשר אותו למסד נתונים, והאחר מעבד ערכת רשומות המבוססת על הקבוצה המוחזרת מהאובייקט.

```
Sub SelectCommand()
Dim cmd1 As Command
Dim rs1 As Recordset, str1 As String
Dim fldLoop As ADODB.Field
' Define and execute command.
    Set cmd1 = New ADODB.Command

    With cmd1
        .ActiveConnection = CurrentProject.Connection
        .CommandText = "SELECT MyTable.* FROM MyTable"
        .CommandType = adCmdText
        .Execute
    End With

' Open and print recordset.
    Set rs1 = New ADODB.Recordset
    rs1.Open cmd1

    Do Until rs1.EOF
        str1 = ""
        For Each fldLoop In rs1.Fields
            str1 = str1 & fldLoop.Value & Chr(9)
        Next fldLoop
        Debug.Print str1
        rs1.MoveNext
    Loop
End Sub
```

חלקה הראשון של השיגרה מצהיר על cmd1 בתור אובייקט Command ולאחר מכן מגדיר שלושה מאפיינים קריטיים של האובייקט. כל פקודה חייבת לכלול מאפיין ActiveConnection כדי לפעול כנגד מסד נתונים. האובייקט Command מסתמך על משפט SQL שייצג את שאילתת הבחירה שלו. תוכל להחליף את שאילתת SQL בשאילה שמורה. משפט Execute מפעיל את שאילתת הבחירה. בתום פעולת השיטה Execute, cmd1 מכיל הפניה אל ערכת רשומות.

חלקה השני של השיגרה פותח אובייקט Recordset המבוסס על cmd1 ומדפיס את קבוצת ההחזרה שחבריה מופרדים בתווי טאב ((chr(9)) בחלון Immediate (מיידי). יכולת הטיפול של השיגרה אינה מוגבלת למספר עמודות ושורות כלשהו.

יצירת ערכת רשומות באמצעות שאילתת פרמטר. הקוד שלפניך הוא דוגמה לשאילתת פרמטר (Parameter Query). גם קוד זה מורכב משני חלקים. שאילתת הפרמטר שבחלק הראשון כוללת שורות קוד ADO אחדות ומשפט SQL בתחביר שונה משל שאילתת הבחירה הקודמת. החלק השני שמדפיס את קבוצת ההחזרה זהה לזה של שאילתת הבחירה הקודמת.

```
Sub ParameterQCommand()
Dim cmd1 As Command
Dim rs1 As Recordset, str1 As String
Dim fldLoop As ADODB.Field
Dim prm1 As ADODB.Parameter, int1 As Integer
' Create and define command.
   Set cmd1 = New ADODB.Command

   With cmd1
      .ActiveConnection = CurrentProject.Connection
      .CommandText = "Parameters [Lowest] Long;" & _
         "SELECT Column1, Column2, Column3 " & _
         "FROM MyTable " & _
         "WHERE Column1>=[Lowest]"
      .CommandType = adCmdText
   End With
' Create and define parameter.
   Set prm1 = cmd1.CreateParameter("[Lowest]", adInteger, adParamInput)
   cmd1.Parameters.Append prm1
   int1 = Trim(InputBox("Lowest value?", "Programming Microsoft Access 2000"))
   prm1.Value = int1

' Run parameter query.
   cmd1.Execute

' Open recordset on cmd1 and print it out.
   Set rs1 = New ADODB.Recordset
   rs1.Open cmd1

   Do Until rs1.EOF
      str1 = ""
      For Each fldLoop In rs1.Fields
         str1 = str1 & fldLoop.Value & Chr(9)
      Next fldLoop
      Debug.Print str1
      rs1.MoveNext
   Loop
End Sub
```

תחביר משפט SQL כולל שורת הצהרה חדשה על Parameters, אשר מציינת את שם הפרמטר וסוג הנתונים שלו. הפסוקית WHERE אף היא צריכה להפנות אל פרמטר אחד או יותר, כדי שהפרמטרים יוכלו לפעול על קבוצת ההחזרה. לא די בתיקוני תחביר אלה במשפט SQL כדי לגרום לשאילתת הפרמטר לפעול – עליך להוסיף את הפרמטר ולצרפו לפקודה באמצעות קוד ADO.

מפעילים את השיטה CreateParameter כדי להוסיף את הפרמטר. הקוד שלעיל מפעיל את השיטה באמצעות שלושה ארגומנטים. הארגומנט הראשון מקצה לפרמטר שם, השני מקצה לו סוג נתונים והארגומנט השלישי מצהיר על כיוון. הקבוע adParamInput הוא למעשה ברירת המחדל שמצהירה על פרמטר קלט לשאילתה. קבועים נוספים מאפשרים להגדיר פרמטרים של פלט, קלט/פלט וערך החזרה. לאחר יצירת פרמטר, עליך לצרף אותו לאוסף Parameters של הפקודה.

לאחר כתיבת הקוד שמוסיף את הפרמטר, עליך להקצות לפרמטר ערך כדי ששאילתת הפרמטר תפעל בצורה תקינה: הקוד שלעיל משתמש בפונקציה InputBox כדי לקבל קלט מהמשתמש. לאחר קבלת הקלט, השיגרה מפעילה את השיטה Execute של האובייקט Command כדי ליצור קבוצת החזרה.

מחיקת רשומות. באפשרותך לנצל את האובייקט Command כדי למחוק, לעדכן ולהוסיף רשומות למקור נתונים. האובייקטים Command מספקים אמצעי תכנותי לתחזוקת מקור נתונים. השגרות DeleteARecord ו-DeleteAllRecords שלהלן אוספות רשומות מתוך מקור נתונים. מייעדים את מקור הנתונים ואת קריטריון בחירת הרשומות באמצעות המשפט DELETE של SQL. תצוגת SQL בחלון השאילתה של Access מאפשר לעצב שאילתה בצורה גרפית ולאחר מכן להעתיק את הקוד למאפיין CommandText של הפקודה. בוודאי תרצה לערוך את קוד SQL מתוך מעצב השאילתות של Access, כדי להסיר סוגריים מיותרים. אם השאילתה שיצרת מתבצעת על טבלה בודדת, תוכל להסיר את תחילית הטבלה שלפני שמות השדות. כפי שניתן לראות, ההבדל בין שתי שאילתות המחיקה מתמקד בתחביר משפט SQL.

```
Sub DeleteARecord()
Dim cmd1 As ADODB.Command

   Set cmd1 = New ADODB.Command

   With cmd1
      .ActiveConnection = CurrentProject.Connection
      .CommandText = "DELETE MyTable.Column1 FROM " & _
         "MyTable WHERE (((MyTable.Column1)=13));"
      .CommandType = adCmdText
      .Execute
   End With

End Sub
```

```
Sub DeleteAllRecords()
Dim cmd1 As ADODB.Command

    Set cmd1 = New ADODB.Command

    With cmd1
        .ActiveConnection = CurrentProject.Connection
        .CommandText = "DELETE MyTable.* FROM MyTable"
        .CommandType = adCmdText
        .Execute
    End With

End Sub
```

הוספת רשומות. בעת פיתוח יישום, ייתכן שתהיה מעוניין ביכולת למחוק את כל הרשומות מתוך טבלה ואחר כך לאפס את תוכנה. השיגרה InsertRecords משתמשת באובייקט Command כדי למלא טבלה בערכים. תוכל להפעיל את השיגרה יחד עם השיגרה DeleteAllRecords כדי לרענן את הטבלה בקבוצת רשומות בסיסית קטנה.

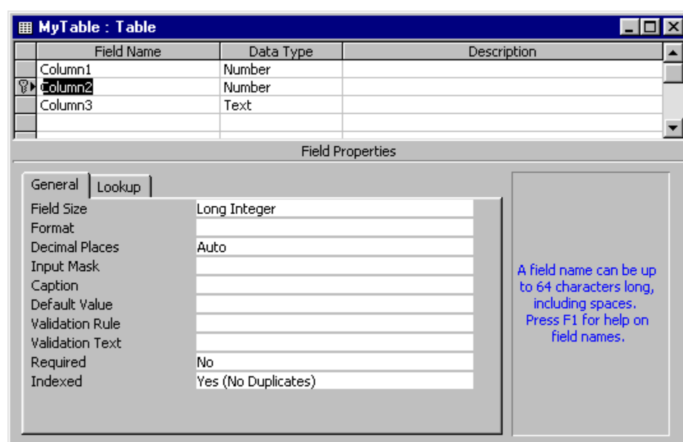
```
Sub InsertRecords()
Dim cmd1 As ADODB.Command

    Set cmd1 = New Command

    With cmd1
        .ActiveConnection = CurrentProject.Connection
        .CommandText = "INSERT INTO MyTable(Column1, " & _
            "Column2, Column3) VALUES (1,2,'3')"
        .CommandType = adCmdText
        .Execute
        .CommandText = "INSERT INTO MyTable(Column1, " & _
            "Column2, Column3) VALUES (4,5,'6')"
        .CommandType = adCmdText
        .Execute
        .CommandText = "INSERT INTO MyTable(Column1, " & _
            "Column2, Column3) VALUES (7,8,'9')"
        .CommandType = adCmdText
        .Execute
        .CommandText = "INSERT INTO MyTable(Column1, " & _
            "Column2, Column3) VALUES (10,11,'12')"
        .CommandType = adCmdText
        .Execute
```

```
        .CommandText = "INSERT INTO MyTable(Column1, " & _
           "Column2, Column3) VALUES (13,14,'15')"
        .CommandType = adCmdText
        .Execute
        .CommandText = "INSERT INTO MyTable(Column1, " & _
           "Column2, Column3) VALUES (16,17,'18')"
        .CommandType = adCmdText
        .Execute
    End With
End Sub
```

השיגרה InsertRecords כוללת אלמנטים כלליים וספציפיים כאחד. האלמנטים הכלליים אינם תלויים בעיצוב טבלה מסוימת. בקוד שלעיל, האלמנטים הספציפיים תואמים לאלה הכלליים של הטבלה MyTable. תרשים 2.7 מציג את MyTable בתצוגת **עיצוב** (Design). הטבלה כוללת שלוש עמודות: Column1, Column2 ו-Column3. שתי העמודות הראשונות מכילות נתונים מסוג Long Integer, והעמודה השלישית מכילה נתונים מסוג Text (בעת הוספת רשומות, יש להתחשב בסוג הנתונים של השדה).



תרשים 2.7: תצוגת **עיצוב** של הטבלה אליה מוסיפים רשומות באמצעות השיגרה InsertRecords

האלמנטים הכלליים של השיגרה InsertRecords משותפים ליישומים נוספים של האובייקט Command. יוצרים הפניה אל אובייקט Command ומגדירים את מאפיין החיבור שלו. לכל שורה ברשומה שברצונך להוסיף, נחוצות שלוש שורות קוד: הגדרת המאפיין CommandText, שמציין את פעולת הפקודה; המאפיין CommandType, שמציין את תבנית ההוראה; והשיטה Execute, שמפעילה את תהליך ההוספה של הרשומה החדשה. תוכל לחזור ולבצע את שלוש שורות הקוד לכל שורה שברצונך להוסיף למקור הנתונים. אם תציין בתור מטרה קבוצת רשומות דינמית שניתן לעדכן אותה, שלבים אלה יכולים להוסיף רשומות לשתי טבלאות או יותר, בו-זמנית.

תחביר משפט CommandText של SQL כולל שלושה מרכיבים (תחביר זה אינו זמין מתוך **תצוגת SQL** של מעצב השאילתות של Access). ראשית הוא משתמש במילת המפתח INSERT INTO. אחרי מילה זו רשום שם מקור הנתונים שאליו ברצונך להוסיף רשומות. שנית, המשפט מנצל את השלב האופציונלי שמציג את שמות השדות שקולטים את הערכים החדשים. אם לא תבצע שלב זה, הערכים שתוסיף בשלב השלישי יתוספו בצורה סדרתית. הדבר עלול להוות בעיה, אם עיצוב מקור הנתונים משתנה במשך הזמן. שלישית, מילת המפתח VALUES מופיעה לפני ערכי השדות של הרשומה החדשה.

עדכון ערכי רשומה. השגרות OddToEven ו-EvenToOdd מעדכנות את ערכי מקור הנתונים של Column1 באמצעות האובייקט Command. תרשים 2.8 מציג את הטבלה מייד לאחר הפעלת השגרות DeleteAllRecords ו-InsertRecords. שים לב כי בתרשים 2.8 הערכים בעמודה Column1 הם זוגיים ואי-זוגיים לחילופין: אם ערך כלשהו בעמודה Column1 הוא אי-זוגי, הערך המקביל בעמודה Column2 הוא זוגי. השגרות מנצלות מידע זה כדי לנהל את תוכן הטבלה.

MyTable : Table

	Column1	Column2	Column3
▶	1	2	3
	4	5	6
	7	8	9
	10	11	12
	13	14	15
	16	17	18
*			

Record: 1 of 6

תרשים 2.8: תצוגת **גיליון נתונים** (Datasheet) של הטבלה אותה מעדכנות השגרות OddToEven ו-EvenToOdd

```
Sub OddToEven()
Dim cmdO2E As ADODB.Command
Dim intRowsChanged As Integer
   Set cmdO2E = New ADODB.Command
   With cmdO2E
      .ActiveConnection = CurrentProject.Connection
      .CommandText = "UPDATE MyTable SET Column1 = " & _
         "Column1+1 WHERE ((-1*(Column1 Mod 2))=True)"
      .CommandType = adCmdText
      .Execute intRowsChanged
      Debug.Print intRowsChanged & " rows were affected."
   End With
End Sub
```

```
Sub EvenToOdd()
Dim cmdE2O As ADODB.Command
    Set cmdE2O = New ADODB.Command
    With cmdE2O
        .ActiveConnection = CurrentProject.Connection
        .CommandText = "UPDATE MyTable SET Column1 = " & _
            "Column1-1 WHERE ((-1*(Column2 Mod 2))=False)"
        .CommandType = adCmdText
        .Execute
    End With
End Sub
```

העיצוב הכולל של שגרות אלו בוודאי מוכר לך. ההבדל המשמעותי ביותר בין הדוגמאות הנוכחיות לאלו שהוצגו קודם לכן טמון בתחביר משפט SQL של המאפיין CommandText. במקרה זה, באפשרותך לחלץ את תבנית התחביר הכללית מתוך מעצב השאילתות של Access. הפסוקית WHERE של השיגרה OddToEven בוחרת רשומות שהערך בעמודה Column1 שלהן הוא אי-זוגי. חלק התחביר הכולל את המילה UPDATE מוסיף 1 לערך כדי להפכו למספר זוגי. השיטה Execute מנצלת אחד מהארגומנטים המוכללים שלה כדי להחזיר את מספר השורות שהשתנו על ידי פקודה. השיטה הפשוטה Print שולחת ערך זה לחלון **Immediate** (מיידי) לצורך הצגתו במסך.

השיגרה EvenToOdd בוחנת את הערך שבעמודה Column2, כדי לקבוע אם יש לחסר 1 מהערך שבעמודה Column1. אם הערך שבעמודה Column2 אינו אי-זוגי, משפט SQL פועל על הערך בעמודה Column1. פעולה זו משחזרת את הערכים שבעמודה Column1 לערכיהם ההתחלתיים, אם EvenToOdd פועלת מייד לאחר השיגרה OddToEven.

האוסף Errors

אוסף זה מאפשר ללכוד חלק מהשגיאות המופיעות ביישום ADO. האוסף גם מחזיר שגיאות מתוך ספק OLE DB. מצב שגיאה יחיד יכול להחזיר שגיאות רבות, שכל אחת מהן גורמת להצבת אובייקט Error חדש באוסף Errors. יש שגיאות הגורמות לסיום התוכנית, אך לא כולן. כל תקלה חדשה מנקה אוטומטית את האוסף Errors כדי שזה יוכל לקלוט את השגיאות הקשורות אליה. שגיאות ADO מסוימות נקלטות באובייקט Err במקום באוסף Errors, אך ניתן להשתמש גם באוסף למטרה זו. האוסף Errors מתאים ביותר לטיפול בשגיאות מבוססות-חיבור שהוחזרו ממסד נתונים מרוחק באמצעות ספק OLE DB שלו.

האובייקט Error של האוסף Errors כולל חמישה מאפיינים שמסייעים לאסוף מידע נוסף המאפשר להגיב עליהם באמצעות לוגיקת התוכנית. המאפיינים Number (מספר) ו-Description (תיאור) מקבילים למאפיינים של האובייקט Err. מאפיינים אלה משלימים זה את זה. המאפיין Number מחזיר מספר ייחודי שמזהה שגיאה, והמאפיין Description מחזיר מחרוזת תיאור קצרה של השגיאה. המאפיין NativeError מספק קוד שגיאה מוגדר של הספק. אם אתה עובד לעיתים קרובות עם ספק מסוים, מאפיין

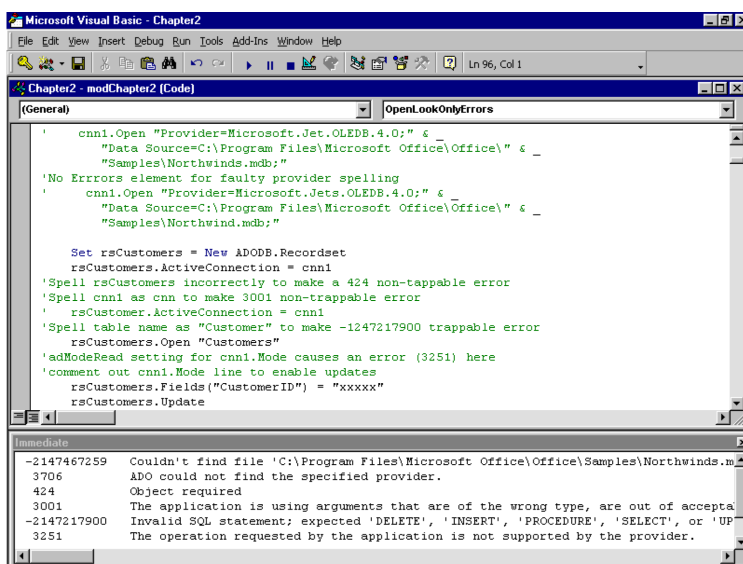
זה עשוי לספק לך מידע שימושי לפתרון השגיאה. המאפיין Source (מקור) נוקב בשם האובייקט או היישום שחולל את השגיאה. המאפיין SQLState יכול להכיל הודעות שגיאת תחביר של משפט SQL, שמקורן בשרת מסד הנתונים שאליו מפנים את הבקשה.

השיגרה OpenLookOnlyErrors שלהלן היא התאמה של שיגרה שהוצגה קודם לכן, אשר מציגה את השפעת המאפיין Mode של האובייקט Connection. הגדרת מאפיין זה כקריאה-בלבד גורמת ליצירת שגיאה בעת שמנסים לעדכן מסד נתונים. מעניין לציין כי השגיאה אינה הופכת לחלק מהאוסף Errors. באפשרותך ללכוד את השגיאה ולהגיב עליה באמצעות האובייקט Err. החבר האחרון באוסף Errors מופיע גם הוא באובייקט Err. לוגיקת לכידת השגיאות שבסוף השיגרה נמנעת מהדפסת שתי שורות בעלות מספר ותיאור זהים.

```
Sub OpenLookOnlyErrors()
Dim cnn1 As New Connection
Dim rsCustomers As Recordset
Dim errLoop As Error, intInErrors As Integer
On Error GoTo LookOnlyTrap
    cnn1.Mode = adModeRead
    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office\Office\" & _
        "Samples\Northwind.mdb;"
' Spell Northwind incorrectly to generate trappable error.
'    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
    "Data Source=C:\Program Files\Microsoft Office\Office\" & _
    "Samples\Northwinds.mdb;"
' No Errors element for faulty provider spelling
'    cnn1.Open "Provider=Microsoft.Jets.OLEDB.4.0;" & _
    "Data Source=C:\Program Files\Microsoft Office\Office\" & _
    "Samples\Northwind.mdb;"
    Set rsCustomers = New ADODB.Recordset
    rsCustomers.ActiveConnection = cnn1
' Spell rsCustomers incorrectly to make a 424 non-trappable error.
' Spell cnn1 as cnn to make 3001 non-trappable error.
'    rsCustomer.ActiveConnection = cnn1
' Spell table name as "Customer" to make -2147217900 trappable error.
    rsCustomers.Open "Customers"
' adModeRead setting for cnn1.Mode causes an error (3251) here.
'Comment out cnn1.Mode line to enable updates.
    rsCustomers.Fields("CustomerID") = "xxxxx"
    rsCustomers.Update
    Debug.Print rsCustomers.Fields("CustomerID")
    rsCustomers.Close
```

```
LookOnlyTrap:
    intInErrors = 0
    For Each errLoop In cnn1.Errors
        Debug.Print errLoop.Number, errLoop.Description
        intInErrors = intInErrors + 1
    Next errLoop
    If intInErrors = 0 Then
        Debug.Print Err.Number, Err.Description
    End If
End Sub
```

השיגרה OpenLookOnlyErrors יוצרת מספר סוגי שגיאה שונים ומנסה לכתוב אל חיבור מסוג קריאה-בלבד. תרשים 2.9 מציג את החלונות Code ו-Immediate של VBE. מקור ההודעות בעלות קודי השגיאה הארוכים והשליליים הוא באוסף Errors. שאר השגיאות הן שגיאות ADO שמדווחות באמצעות האובייקט Err. רק לשתי שגיאות יש מספרים ארוכים ושליליים מהסוג המאפיין את השגיאות שבאוסף Errors. שאר השגיאות הן שגיאות ADO שניתן להגיע אליהן באמצעות האובייקט Err. שגיאה אחת מתוך מחרוזת החיבור (שגיאה מספר 3706) לא הוצגה עדיין באמצעות האוסף Errors. זאת בצירוף העובדה שהחבר האחרון באוסף Errors מופיע באובייקט Err, מהווים יתרון בכל הקשור ללכידת שגיאות באמצעות האובייקט Err. מנגנון זה פועל גם בשגיאות VBA.



תרשים 2.9: החלונות Code ו- Immediate של VBE מציגים קודי שגיאה של שגיאות טיפוסיות



הערה:
באפשרותך להוסיף משפט Option Explicit באזור ההצהרות הכלליות של מודול, כדי לבטל את האפשרות של שגיאות מסוימות, כגון הפניות לאובייקטים שאינם קיימים.

השיגרה LoopToUsingErrors שלפניך מספקת מספר גישות חלופיות לטיפול בשגיאות באמצעות האובייקט Err. השיגרה יוצרת שגיאה אחת וכוללת הערה המפרטת את שינויי הקוד שיש לבצע כדי ליצור שגיאה נוספת. השיגרה אף מגיבה באופן ייחודי לשתי שגיאות, וכוללת הליך כללי לטיפול בכל יתר השגיאות:

- במקרה של שגיאה 3251, השיגרה משנה את סוג הנעילה, כך שניתן לעדכן את ערכת הרשומות. שגיאה זו מתרחשת מכיון שסוג הנעילה שגוי. לתיקון הבעיה, קוד הטיפול בשגיאה סוגר את ערכת הרשומות הישנה, מאפס את המאפיין LockType ופותח מחדש את אובייקט ערכת הרשומות.
- במקרה של שגיאה 424, השיגרה אינה מנסה לתקן את השגיאה, אלא מתריעה בפני המשתמש על סיבת הבעיה האפשרית. שגיאה זו מתרחשת כאשר מפעילים שיטה או מגדירים מאפיין כנגד משתנה שלא הוצהר כאובייקט. לדוגמה, שגיאת הקלדה עשויה לגרום לבעיה כזו.
- אם מספר השגיאה שונה מ-3251 או 424, השיגרה מדפיסה את מאפייני המספר והתיאור של האובייקט Err.

```
Sub LoopToUsingErrors()

On Error GoTo DErrorsTrap

Dim cnn1 As Connection
Dim rsMyTable As Recordset

    Set cnn1 = New ADODB.Connection
    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office\Office\" & _
        "Samples\Northwind.mdb;"

    Set rsMyTable = New ADODB.Recordset

' Open recordset with defaults.
OpenRSMyTable:
    rsMyTable.Open "MyTable", cnn1
```

```
' Loop through recordset.
   Do Until rsMyTable.EOF

' Make 424 error by using next instead of preceding line.
'    Do Until rsMyTables.EOF
   If rsMyTable.Fields(0) = 4 Then

' This line makes 3251 error because recordset is read-only.
      rsMyTable.Fields(0) = 88
   Else
      Debug.Print rsMyTable.Fields(0), rsMyTable.Fields(1)
   End If

   rsMyTable.MoveNext
   Loop

   rsMyTable.Close

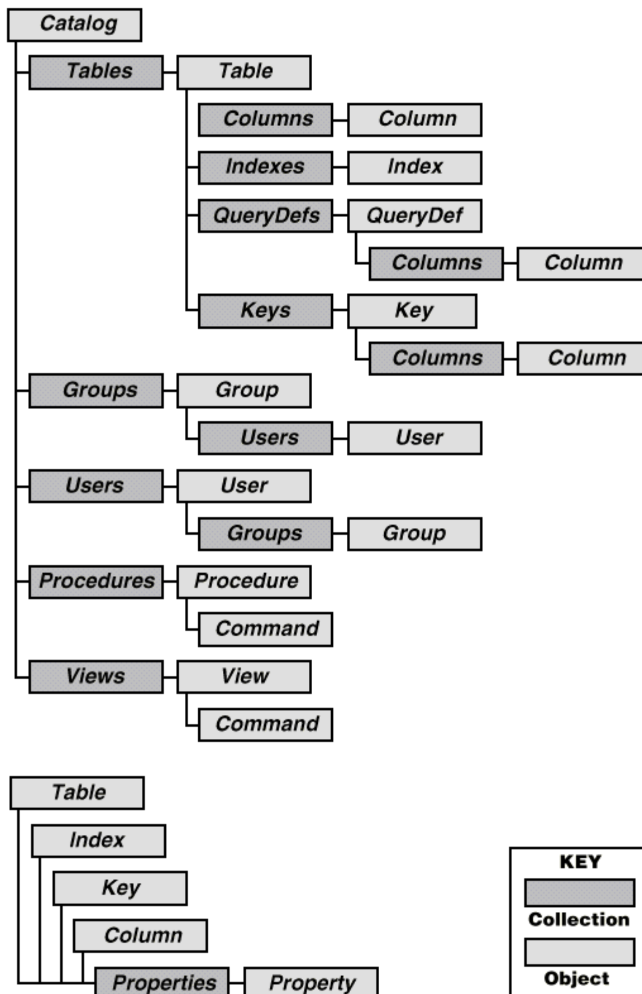
ErrorsExit:
   Exit Sub

DErrorsTrap:
   If Err.Number = 3251 Then
      MsgBox "OLEDB Provider does not support operation. " & _
         "Find another way to get the job done or get a new " & _
         "OLEDB Provider. Error happened in LoopToDeleteErrors."
      Debug.Print rsMyTable.LockType
      rsMyTable.Close
      rsMyTable.LockType = adLockOptimistic
      Resume OpenRSMyTable
   ElseIf Err.Number = 424 Then
      MsgBox "The code tried to do something requiring an " & _
         "object, such as set a property or invoke a method, " & _
         "but the code did not have an object. Check spelling."
   Else
      MsgBox "Check Immediate window for error # and desc."
         Debug.Print Err.Number, Err.Description
   End If
   Resume ErrorsExit

End Sub
```

הספריה ADOX

הספריה ADOX תומכת במשימות סכימה ואבטחה. באפשרותך לנצל ספריה זו כדי לשנות את הארכיטקטורה של עיצוב מסד הנתונים של היישום שלך. כל האובייקטים, למעט האובייקט Catalog, כוללים אוספים תואמים. אוספים אלה משמשים להוספה וארגון אובייקטים חדשים בקטלוג. אובייקטים נבחרים, כגון טבלאות, מפתחות אינדקס ועמודות, כוללים אוספים מסוג Properties (מאפיינים). אוספים אלה משמשים לניהול פעולת האובייקטים ביישום. מנהלים את האוספים Users (משתמשים) ו-Groups (קבוצות) כדי לשלוט בהרשאות של אובייקטים נוספים של ADOX, כגון טבלאות, תצוגות ושגרות. תרשים 2.10 מציג מבט כללי על הספריה ADOX.



תרשים 2.10: ספריית האובייקטים ADOX

הספריה ADOX היא שלוחה של הספריה ADODB. הספק Jet ADO תומך ב-ADOX באופן מלא. ספקי מסדי נתונים אחרים יכולים ליישם את רכיביו בצורה סלקטיבית, וניתן גם להשתמש בשתי הספריות יחדיו כדי לבנות יישומים. לדוגמה, באפשרותך להרכיב אובייקטים של פקודה בעזרת הספריה ADODB ואחר לשמור אותם כשגרות בספריה ADOX. לחילופין, תוכל לבדוק זמינות טבלה בטרם תבסס עליה ערכת רשומות. אם הטבלה אינה קיימת, תוכל להוסיפה ולאכלס אותה בערכים. יכולת הספריה ADOX להגדיר מבני נתונים חדשים הופכת אותה לחלופה של SQL DDL.

האובייקט Catalog

אובייקט זה הוא המכולה ברמה הגבוהה ביותר של הספריה ADOX. חבריו מגדירים את הסכימה ואת מודל האבטחה של מסד נתונים. המאפיין ActiveConnection של האובייקט מגדיר את החיבור שאליו שייך הקטלוג. האובייקט Connection הוא מסד הנתונים. האובייקט Catalog הוא מכולת הטבלאות, התצוגות, השגרות, המשתמשים והקבוצות בתוך חיבור או מסד נתונים. המאפיין Name (שם) של קטלוג הוא מסוג קריאה-בלבד. מגדירים אותו בעת ההצהרה על הקטלוג. משתמשים בשיטה Create של האובייקט Catalog כדי להקצות חיבור ומקור נתונים לקטלוג, כך שניתן יהיה לפתוח בו-זמנית מסד נתונים חדש ולקבל גישה לקטלוג שלו.

דוגמאות הקוד שלפניך מציגות את האובייקט Catalog בשלושה הקשרים טיפוסיים. השיגרה CatCon פותחת את מסד הנתונים Northwind ומספקת ליישום הנוכחי גישה באמצעות קוד למבנה מסד הנתונים. שים לב שיש צורך באובייקט Connection עבור הקטלוג, כדי ש-ADO תדע איזה קטלוג עליה להפוך לזמין. מקצים את האובייקט Connection לקטלוג באמצעות המאפיין ActiveConnection. כש-ADO יודעת לאיזה מסד נתונים עליה להפנות בעזרת הקטלוג, יש לך גישה באמצעות קוד לתוכן הקטלוג. שולטים בגישה באמצעות טכניקות מסד נתונים ואבטחה ברמת-משתמש.

```
Sub CatCon()
Dim cnn1 As New Connection
Dim cat1 As New Catalog
Dim proc1 As Procedure

   cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=C:\Program Files\Microsoft Office\Office\" & _
      "Samples\Northwind.mdb;"
   Set cat1.ActiveConnection = cnn1
   Debug.Print cat1.Tables(0).Name

End Sub
```

```
Sub CatCon2()
Dim cnn1 As New Connection
Dim cat1 As New Catalog
Dim proc1 As Procedure

' Open the catalog of this database.
' Print a range of selected collection information.
    Set cnn1 = CurrentProject.Connection
    Set cat1.ActiveConnection = cnn1
    Debug.Print cat1.Tables(1).Name
    Debug.Print cat1.Views(0).Name
    Debug.Print cat1.Procedures(0).Name
    Debug.Print cat1.Users(0).Name
    Debug.Print cat1.Groups(0).Name

End Sub

Sub CatCon3()
Dim cat1 As New ADOX.Catalog

' Open the catalog to a new database.
    cat1.Create "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=c:\My Documents\NewDB.mdb"
    Debug.Print cat1.Tables(0).Name

End Sub
```

השיגרה CatCon2 מציגה חבר מכל אוסף של האובייקט Catalog במסד הנתונים הנוכחי. תוכל להפנות לתוכן טבלה באמצעות הפניה לספריה ADODB, אך עליך להשתמש בהפנייה ל-ADOX כדי לעבור בלולאה על האוסף Columns (עמודות). האוספים Columns ו-Tables (טבלאות) קיימים רק בספריה ADOX.



הערה:

עליך לכפות דרישת כניסה (logon) בטרם תוכל להציג את חברי האוספים Users ו-Groups. כל ניסיון להדפיס אותם ללא כניסה עלול ליצור שגיאת זמן ריצה. תוכל לכפות דרישת כניסה על ידי קביעת סיסמה עבור המשתמש Admin.

השיגרה CatCon3 יוצרת מסד נתונים חדש ומציגה את הקטלוג שלו באותו שלב. שים לב כי אתה מפעיל את השיטה Create על האובייקט Catalog. השיטה מקבלת מחרוזת חיבור כארגומנט. אם הקובץ המצוין במחרוזת החיבור כבר קיים, CatCon3 תיצור שגיאת זמן ריצה. כשהקובץ פועל בהצלחה, הוא מדפיס את שם אחת מטבלאות המערכת, מכיון שלא קיימים קבצי משתמש זמינים כלשהם מייד לאחר יצירת מסד נתונים.

באפשרותך לנצל את האובייקט Catalog כדי למספר חברים של אוסף כלשהו מהאוספים שהוא כולל. השיגרה שלפניך ממספרת את חברי האוסף Views (תצוגות), תהליך המקביל להגדרת קבוצת כל שאילתות החזרת-שורות שאינן מבוססות על פרמטרים בעלי מסדי נתונים Jet. בעת שימוש במסד נתונים SQL Server, התצוגות זמינות במפורש מתוך מכולת מסד הנתונים של Access 2000.

```
Sub ListViews()
Dim cnn1 As New Connection
Dim cat1 As New Catalog
Dim view1 As View
    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office\Office\" & _
        "Samples\Northwind.mdb;"
    Set cat1.ActiveConnection = cnn1
    For Each view1 In cat1.Views
        Debug.Print view1.Name
    Next view1
End Sub
```

האוסף Views זמין מתוך הקטלוג עבור החיבור. כדי למספר את התצוגות, עליך להצהיר על תצוגה אחת שהקוד שלך ינצל כדי להפנות אל כל חבר באוסף Views במהלך מעבר הלולאה על האוסף. הדפסת המאפיין Name (שם) של כל תצוגה מהווה רישום מצאי של אובייקטי View שבקטלוג.

האובייקט Table

אובייקט זה הוא חבר באוסף Tables, כאשר זה הוא חבר באובייקט Catalog. כל אובייקט Table כולל מאפיין Name ומאפיין Type. אובייקט Table יכול להיות טבלה רגילה בתוך מסד הנתונים הנוכחי, או טבלה מקושרת ממקורות נתונים ODBC וכאלה שאינם ODBC. האובייקט יכול להיות אפילו תצוגה. ערך המאפיין Type כולל גם שני סוגים של טבלאות מערכת: טבלאות מערכת Jet וטבלת מערכת Access.

ערכי מאפיין Type של אובייקט Table	תיאור
ACCESS TABLE	טבלת מערכת של Access
LINK	טבלה מקושרת מתוך מקור נתונים שאינו ODBC
PASS-THROUGH	טבלה מקושרת דרך מקור נתונים ODBC
SYSTEM TABLE	טבלת מערכת Jet
TABLE	טבלה שפותחה באמצעות יישום או עבורו
VIEW	טבלה מתוך שאילתת החזרת שורות לא-פרמטרית

אובייקט Table בספריה ADOX יכול להכיל עד שלושה אוספים: Columns, Indexes ו-Keys.

האובייקט Column

האוסף Columns מיועד לטבלאות, מפתחות ואינדקסים. אובייקט מסוג Column דומה במידת-מה לאובייקט מסוג Field בספריה ADODB. עמודה בטבלה היא קבוצת כל הנתונים המתייחסים לתכונה מוגדרת של היישות שמיוצגת באמצעות הטבלה. לאובייקט Column יש כמה מאפיינים:

- **Name**. מאפיין זה הוא שם העמודה.
- **Type**. המאפיין מציין את סוג הנתונים שבעמודה. כל הנתונים מסוג אחד.
- **Attribute**. מאפיין זה מתאר את האפיונים של עמודה. לעמודה שני אפיונים: היכולת להכיל ערכי Null והיותה בעלת אורך קבוע.
- **DefineSize**. מאפיין זה מציין את האורך המירבי של ערך בעמודה.
- **Precision** ו-**NumericScale**. מאפיינים אלה משמשים באופן בלעדי עבור שדות מספריים, כגון שדות מסוג integer, מטבע וערכי נקודה צפה (floating-point). המאפיין Precision מייצג את מספר הספרות הכולל שמשמשות להעברת ערך העמודה. NumericScale קובע את מספר הספרות העשרוניות שיבטאו את הערך.

כאשר אובייקט מסוג Column הוא אינדקס, מאפיינים אחרים כגון SortOrder ו-RelatedColumn יהיו זמינים אף הם.



הערה:

המאפיין NumericScale עלול לגרום לבלבול. לדוגמה, ערכים מסוג Currency מבוטאים באמצעות ארבע ספרות מימין לנקודה העשרונית, אך המאפיין NumericScale שלהם הוא 0, כיון ש-Access מאחסן נתוני Currency בתור scaled integers. בעת שינוי הגדרת Scale של עמודה שמשתמשת בסוג הנתונים Decimal בתצוגת **עיצוב** (Design) של טבלה, המאפיין NumericScale של העמודה מתאים את עצמו לפי הנסיבות.

האובייקט Index

אובייקט זה מגדיר אינדקסים עבור טבלה. לאובייקט חמישה מאפיינים: Clustered, IndexNulls, Name, PrimaryKey ו-Unique. לאחר הוספת האינדקס, כל המאפיינים למעט Name, הם מסוג קריאה-בלבד. המאפיין Name הוא שם האינדקס. יש שלושה מאפיינים נוספים מסוג Boolean, ותפקידיהם (בהתאמה): לציין אם האינדקס **מקובץ-באשכולות** (clustered) (אינדקס נקרא מקובץ-באשכולות, כאשר הסדר הפיסי של השורות תואם לסדר ה׳מאונדקס׳ של השורות) מפתח ראשי (PrimaryKey) או ייחודי (Unique).

המאפיין IndexNulls יכול לקבל ערך אחד מתוך שלושה ערכים. במקרה שהאינדקס מכיל ערך Null עבור עמודה, הגדרתו בתור adIndexNullsDisallow גורם לכשלון בבניית Index. אם מציבים ב-IndexNulls את הקבוע adIndexNullsIgnore, אפשר לבנות את האינדקס גם אם הוא מכיל ערך Null, אך המאפיין **התעלם מ-Nulls** (Ignore Nulls) יקבל ערך **כן** (Yes) (בחלון **אינדקסים** (Indexes) שבממשק המשתמש). גם הערך adIndexNullsIsIgnoreAny יוצר את האינדקס כשזה מכיל ערך Null, במקרה זה המאפיין **התעלם מ-Nulls** מקבל ערך **לא** (No).

האובייקט Key

אובייקט זה מגלם את התנהגותם של מפתחות זרים (foreign keys) באמצעות מאפייניו. כמובן, המאפיין Name הוא שם המפתח. המאפיין RelatedTable מציין את הטבלה שעליה מצביע מפתח זר. המאפיינים DeleteRule ו-UpdateRule קובעים מה יקרה כאשר מפתח ראשי נמחק או מתעדכן. המאפיין Type הוא סוג המפתח, ולו שלוש אפשרויות: adKeyForeign עבור מפתחות זרים, adKeyUnique עבור מפתחות ייחודיים ו-adKeyPrimary עבור מפתחות ראשיים.

מספור טבלאות. אחת הדרכים הקלות ביותר להתחיל לטפל בטבלאות היא למספר אותן. השיגרה ListTables שלפניך מציגה כיצד משבצים את הטבלאות במסד הנתונים Northwind. ההצהרות הופכות את cat1 למופע של האובייקט Catalog, ואת tbl1 למופע של אובייקט Table. לאחר מכן, השיגרה מציבה את מסד הנתונים Northwind ואת ספק Jet 4 במאפיין ActiveConnection של הקטלוג. הלולאה שלפניך מזהה את שם הטבלה הארוך ביותר בקטלוג. הקטע האחרון מדפיס את שמות הטבלאות בחלון **Immediate**. נוסחת מחרוזת מוסיפה תווי רווח בסוף שמות הטבלאות כך שמספר התווים בכל השמות יהיה קבוע. הקוד מצרף את סוג הטבלה.

```
Sub ListTables()
Dim cat1 As New ADOX.Catalog
Dim tbl1 As ADOX.Table
Dim intMaxLength As Integer

' Specify active connection for the Catalog .
   cat1.ActiveConnection = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=C:\Program Files\Microsoft Office\Office\" & _
      "Samples\Northwind.mdb;"

' Find longest table name.
   intMaxLength = 0
   For Each tbl1 In cat1.Tables
      If Len(tbl1.Name) > intMaxLength Then intMaxLength = Len(tbl1.Name)
   Next tbl1
```

```
' Print table names to Immediate window.
    intMaxLength = intMaxLength + 2
    For Each tbl1 In cat1.Tables
        strName = tbl1.Name
        strFiller = String(intMaxLength - Len(tbl1.Name), " ")
        Debug.Print strName & strFiller & tbl1.Type
    Next tbl1
End Sub
```

```
Immediate
Categories                        TABLE
Category Sales for 1995           VIEW
Current Product List              VIEW
Customers                         TABLE
Employees                         TABLE
MSysAccessObjects                 ACCESS TABLE
MSysACEs                          SYSTEM TABLE
MSysObjects                       SYSTEM TABLE
MSysQueries                       SYSTEM TABLE
MSysRelationships                 SYSTEM TABLE
MyTable                           TABLE
Order Details                     TABLE
Order Details Extended            VIEW
Order Subtotals                   VIEW
Orders                            TABLE
Product Sales for 1995            VIEW
Products                          TABLE
Products Above Average Price      VIEW
qryForSource                      VIEW
Quarterly Orders                  VIEW
Sales by Category                 VIEW
Shippers                          TABLE
Suppliers                         TABLE
Ten Most Expensive Products       VIEW
```

תרשים 2.11: פלט השיגרה ListTables

תרשים 2.11 מציג את פלט השיגרה ListTables. שים לב שהפלט מופיע בשתי עמודות, וכל אחת מציגה שם טבלה. ליד השם מופיע המאפיין Type של הטבלה. כזכור, האובייקט Table כולל שישה סוגים. ארבעה מהם מופיעים בתרשים.

שדות מספור. השגרות ListTableTypeColumns ו-ColumnType שלפניך הן דוגמאות משוכללות יותר, שמטפלות במאפייני הטבלה ובהיררכיה שלה. כש-ListTableTypeColumns מאתרת אובייקט מסוג Table שהמאפיין Table שלו מוגדר בתור TABLE, היא מציגה את שמות העמודות ואת סוגי כל עמודות הטבלה. אלה מופיעים תחת שם הטבלה ומספר העמודות שלה. כל שם עמודה מופיע לצד הקבוע ColumnType. תרשים 2.12 מציג קטע מהתדפיס. זו דוגמה עשירה, כיון שהיא מטפלת בכמה סוגי אובייקטים מסוג Table ובמאפיינים רבים, ומשום שהאובייקטים והמאפיינים נלקחו מנקודות שונות בהיררכית מודל האובייקט של ADOX.

```
Immediate
Categories                    4
     CategoryID          adInteger
     CategoryName        adWChar
     Description         adLongVarWChar
     Picture             adLongVarBinary
Customers                     11
     Address             adWChar
     City                adWChar
     CompanyName         adWChar
     ContactName         adWChar
     ContactTitle        adWChar
     Country             adWChar
     CustomerID          adWChar
     Fax                 adWChar
     Phone               adWChar
     PostalCode          adWChar
     Region              adWChar
Employees                     17
     Address             adWChar
     BirthDate           adDate
     City                adWChar
     Country             adWChar
     EmployeeID          adInteger
     Extension           adWChar
     FirstName           adWChar
     HireDate            adDate
     HomePhone           adWChar
     LastName            adWChar
     Notes               adLongVarWChar
```

תרשים 2.12: פלט השגרות ListTableTypeColumns ו-ColumnType

```
Sub ListTableTypeColumns()
Dim cat1 As New ADOX.Catalog
Dim tbl1 As ADOX.Table
Dim col1 As ADOX.Column
   cat1.ActiveConnection = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=C:\Program Files\Microsoft Office\" & _
      "Office\Samples\Northwind.mdb;"
   For Each tbl1 In cat1.Tables
      If tbl1.Type = "TABLE" Then
         strName = tbl1.Name
         strFiller = String(30 - Len(tbl1.Name), " ")
         Debug.Print strName & strFiller & tbl1.Columns.Count
         For Each col1 In tbl1.Columns
            strFiller = String(20 - Len(col1.Name), " ")
            Debug.Print String(5, " ") & col1.Name & _
            strFiller & ColumnType(col1.Type)
         Next col1
      End If
   Next tbl1
End Sub
```

```
Public Function ColumnType(intType As Integer) As String
    Select Case intType
        Case adVarWChar
            ColumnType = "adVarWChar"
        Case adCurrency
            ColumnType = "adCurrency"
        Case adInteger
            ColumnType = "adInteger"
        Case adDate
            ColumnType = "adDate"
        Case adWChar
            ColumnType = "adWChar"
        Case adLongVarWChar
            ColumnType = "adLongVarWChar"
        Case adLongVarBinary
            ColumnType = "adLongVarBinary"
        Case adBoolean
            ColumnType = "adBoolean"
        Case adSmallInt
            ColumnType = "adSmallInt"
        Case Else
            ColumnType = CStr(intType)
    End Select
End Function
```

ההצהרות כוללות קטלוג, טבלה ועמודה מתוך הספריה ADOX. הצהרת הקטלוג יוצרת מופע של האובייקט שלו, אך השתיים הנותרות משמשות כהפניות חבר בתוך אוסף. מייד לאחר ההצהרות, השיגרה מקצה מחרוזת חיבור אל המאפיין ActiveConnection של האובייקט Catalog.

שתי לולאות For מבצעות חיפוש בטבלאות ובעמודות שלהן. הלולאה החיצונית מחפשת טבלאות שהמאפיין Table שלהן הוגדר בתור TABLE. ברגע שנמצא כזה, היא מדפיסה את השם ומספר העמודה של אותה טבלה. לולאת For הפנימית ממספרת את עמודות הטבלה ומטפלת בעיצוב הפלט. הלולאה קוראת לפונקציה ColumnType שמחזירה מחרוזת תווים. המחרוזת מייצגת את הקבוע שמגדיר את המאפיין Type. הפונקציה מצטיינת בחסינותה, ולכן תוכל לאמץ אותה לביצוע מטלות המרה. אם הפונקציה נתקלת בסוג בלתי מזוהה, היא ממירה את ערך הסוג למחרוזת. הדבר מסייע לחזור לפונקציה ולהוסיף לה מפרט פענוח חדש.

יצירת טבלאות. לספריה ADOX חשיבות מיוחדת, כיון שהיא מאפשרת ליצור טבלאות באמצעות קוד, יחד עם האינדקסים והמפתחות שלהן.

אם נוח לך להוסיף טבלאות בתצוגת **עיצוב** (Design), עליך להשקיע מעט יותר כדי ללמוד לעשות זאת באמצעות קוד. מגדירים את הטבלה ולאחר מכן מוסיפים אותה לאוסף Tables של הקטלוג. הגדרת טבלה דורשת לוגיקה מסוג דומה. מצהירים תחילה על משתנה אובייקט טבלה ולאחר מכן מצרפים עמודות למשתנה האובייקט. בעת הוספת העמודות, הקוד יכול להקצות סוגי נתונים ומאפייני עמודה נוספים. בסיום הגדרת הטבלה, ניתן למלא אותה בנתונים. באחד הסעיפים הקודמים בפרק, הצגנו שתי גישות לטיפול בנושא (שימוש בקוד SQL ובשיטה AddNew של ערכת הרשומות). בפרק 3 נבחן את הנושא בהרחבה.

הוספת אינדקס. השיגרה AddIndex שלהלן מדגימה כיצד מוסיפים אינדקס לטבלה באמצעות קוד. כמו בתהליך יצירת טבלה, משתמשים בשיטה Append פעמיים. ראשית, הוסף עמודת נתונים אחת או יותר לאינדקס שיצרת. לאחר מכן הגדר את המאפיין Name של האובייקט Index. באפשרותך גם להגדיר מאפיינים נוספים. שורות ההערה בשיגרה שלפניך מציגות הוראות להגדרת מאפיינים אופציונליים אחדים ומתארות את ההשפעה של מתן ערכים למאפיינים. לאחר שתסיים להגדיר את מפרט האינדקס, הפעל את השיטה Append פעם נוספת, אך הפעם השתמש בה כדי להוסיף את האינדקס החדש לאוסף Indexes של הטבלה.

```
Sub AddIndex()
Dim cat1 As New ADOX.Catalog
Dim tbl1 As New ADOX.Table
Dim idx1 As New ADOX.Index
    cat1.ActiveConnection = CurrentProject.Connection

    Set tbl1 = cat1.Tables("MyTable")
    idx1.Name = "MyFirstIndex"
    idx1.Columns.Append ("Column1")
' Rules and syntax for setting IndexNulls property
' Does not create index if field contains Nulls; yields error.
'    idx1.IndexNulls = adIndexNullsDisallow
' Sets Ignore Nulls index property to Yes -- creates index.
'    idx1.IndexNulls = adIndexNullsIgnore
' Sets Ignore Nulls index property to No -- creates index.
'    idx1.IndexNulls = adIndexNullsIgnoreAny

' If you want to set the PrimaryKey property
'    idx1.PrimaryKey = True
' you must also set the Unique property for the
' primaryKey property to take effect and avoid an error.
'    idx1.Unique = True
    tbl1.Indexes.Append idx1
End Sub
```

השיגרה PKErrorCatcher שלפניך מבצעת לכידת שגיאות בעת יצירת מפתח ראשי חדש בטבלה. הרעיון הכללי של שיטת לכידת השגיאות מיושם גם באינדקסים נוספים ובמפתחות זרים. מנגנון לכידת השגיאות מטפל מפורשות בשני סוגי שגיאה ומסוגל ללכוד גם שגיאות מסוגים אחרים.

```
Sub PKErrorCatcher()
On Error GoTo PKErrorCatcherTrap
Dim catMyCat As New ADOX.Catalog
Dim tblMyTable As New ADOX.Table
Dim idxMyIndex As New ADOX.Index
Dim colMyCol As New ADOX.Column
Dim iNumber As Integer
catMyCat.ActiveConnection = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office\Office\" & _
        "Samples\Northwind.mdb;"
Set tblMyTable = catMyCat.Tables("MyTable")
PKErrorCatcherTry:
    With idxMyIndex
        .Name = "MyPrimaryKey"
        .PrimaryKey = True
        .Unique = True
        .IndexNulls = adIndexNullsDisallow
    End With
idxMyIndex.Columns.Append "Column1"
    tblMyTable.Indexes.Append idxMyIndex
Set catMyCat = Nothing
PKErrorCatcherExit:
    Exit Sub
PKErrorCatcherTrap:
    If Err.Number = -2147217856 Then
        MsgBox "Table currently in use."
    ElseIf Err.Number = -2147467259 Then
        For Each idxMyIndex In tblMyTable.Indexes
            If idxMyIndex.PrimaryKey = True Then
                tblMyTable.Indexes.Delete (iNumber)
                Resume PKErrorCatcherTry
            End If
            iNumber = iNumber + 1
        Next idxMyIndex
    Else
        MsgBox "Error #" & Err.Number & ": " & Err.Description
    End If
    Resume PKErrorCatcherExit
End Sub
```

השיגרה פותחת בהצהרה על האובייקטים החדשים Catalog, Table ו-Index (מפתח ראשי הוא אינדקס שהמאפיינים Unique ו-PrimaryKey שלו קיבלו את הערך True). חלק הארי של PKErrorCatcher מגדיר את המאפיינים הדרושים, מצרף עמודה לאינדקס ולאחר מכן מצרף את האינדקס לטבלה. קיימים לפחות שני מצבים בהם הוראות פשוטות אלו ייכשלו בזמן ריצה. האחד, הטבלה עלולה להיות פתוחה. ADO מקצה למצב זה את מספר השגיאה **-2147217856**. כשמנגנון לכידת השגיאות מזהה מספר זה, הוא מציג הודעה למשתמש, בה נאמר כי הטבלה נמצאת כרגע בשימוש. מצב נוסף, הניסיון לצרף מפתח ראשי חדש עלול להיכשל, אם מוגדר כבר אחד כזה עבור הטבלה. השיגרה מזהה את המפתח הראשי הישן ומנסה ליצור מפתח ראשי חדש. כיון שלא קיים יותר מפתח ראשי כלשהו, השיגרה אינה יכולה להיכשל שנית מאותה סיבה.

מתן ערכים לשדה autoincrement. היישום Access 2000 הוא הגירסה הראשונה של Access שמאפשרת למפתחים להגדיר את ערכי ההתחלה והפסיעה של עמודות autoincrement (מספור אוטומטי בהפרשים קבועים). באפשרותך להגדיר ערכים אלה באמצעות קוד, על ידי יצירת טבלה באמצעות משפטי SQL של Jet 4. הפקודה CREATE TABLE משמשת ליצירת המבנה הכללי של הטבלה, וסוג הנתונים IDENTITY של Jet SQL משמש לטיפול בשדה autoincrement. סוג הנתונים IDENTITY כולל ערך התחלתי וערך פסיעה שמאפשרים לתת ערך התחלתי לשדה autoincrement, ואת ערך ההגדלה שלו בכל רשומה חדשה. השיגרה SetStartAndStep שלפניך מנצלת את הטכנולוגיה החדשה.

```
Sub SetStartAndStep()
Dim cnn1 As Connection
Dim cmd1 As Command
Dim tbl1 As New Table
    Set cnn1 = CurrentProject.Connection
    Set cmd1 = New ADODB.Command

    With cmd1
        .ActiveConnection = cnn1
' First create a table with two columns.
' Make one column an Identity column.
' Set its start value first, and its step value second.
        .CommandType = adCmdText
        .CommandText = "CREATE TABLE Contacts (ContactID " & _
            "IDENTITY(2,4),ContactName Char)"
        .Execute
' After creating the table with the autoincrement/identity
' column you should add data.
        .CommandText = "INSERT INTO Contacts(ContactName) " & _
            "Values ('Kevin Mineweaser')"
        .CommandType = adCmdText
        .Execute
```

```
        .CommandText = "INSERT INTO Contacts(ContactName) " & _
            "Values ('Mike Gilbert')"
        .CommandType = adCmdText
        .Execute
        .CommandText = "INSERT INTO Contacts(ContactName) " & _
            "Values ('Neil Charney')"
        .CommandType = adCmdText
        .Execute
    End With

End Sub
```

הקוד פותח ביצירת טבלה. למרות שהשיגרה אינה כוללת קוד ללכידת שגיאות, עליך להבטיח כי הטבלה שאתה יוצר אינה קיימת כבר, או כוללת מנגנון ללכידת שגיאות שמוחק את הטבלה במידה שזו אכן קיימת. מגדירים את סוג הנתונים IDENTITY במסגרת המשפט CREATE TABLE. נותנים ערך התחלתי לארגומנט הראשון של השדה autoincrement וערך פסיעה לארגומנט השני. הקוד מציין כי הקשרים (contacts) יתחילו בערך 2 ויגדלו ב-4 עבור כל ערך חדש. תרשים 2.13 מציג את פלט השיגרה SetStartAndStep. שים לב שהשדה ContactID מקבל ערך התחלתי 2 וגדל בפסיעות של 4. אלו הן ההגדרות של סוג הנתונים IDENTITY המצוין בשיגרה.

Contacts : Table

	ContactID	ContactName
▶	2	Kevin Mineweaser
	6	Mike Gilbert
	10	Neil Charney
*	(AutoNumber)	

Record: 1 of 3

תרשים 2.13: פלט השיגרה SetStartAndStep

האובייקט View

אובייקט זה הוא שאילתת החזרת שורות שאינה מחייבת פרמטרים כלשהם. כשהיישום שומר את סוגי השאילתות האחרים, הוא מאחסן אותם כחברים באוסף Procedures. באפשרותך לטפל בחברי האוסף Views באמצעות האובייקט Command והאוספים Views ו-Tables. האובייקט View כולל שני מאפיינים קריטיים לנושא בו אנו עוסקים: המאפיין Name הוא שם התצוגה, והמאפיין Command מאפשר להגיע למשפט SQL שעליו מבוססת התצוגה.

בסעיף זה ניצור תצוגה באמצעות השיגרה MakeAView וגם נערוך את משפט SQL שעליו מבוססת תצוגה קיימת באמצעות השיגרה ChangeAView. השגרות MakeAView ו-ChangeAView נעזרות שתיהן בשיגרה ViewAView. הקוד מדפיס בחלון Immediate שלושה שדות מתוך כל תצוגה.

יצירת תצוגה. השיגרה MakeAView שלפניך מוסיפה תצוגה חדשה על ידי יצירת אובייקט Command שמייצג את התצוגה, ומצרפת את האובייקט Command לאוסף Views. MakeAView זקוקה לחיבור אל מסד נתונים. האובייקט cnn1 Connection פותח קישור אל מסד הנתונים Northwind. לאחר מכן, קוד היישום מציב את ערך המאפיין ActiveConnection של האובייקט Command שבדוגמה, כ-cnn1. כעת מגיע השלב הקריטי: הקוד מגדיר את המאפיין CommandText של cmd1. המאפיין CommandText מכיל את משפט SQL שמגדיר את התצוגה. משפט SQL של MakeAView בונה את התצוגה כדי להציג שם ושם משפחה ואחר כך מספר שלוחת הטלפון. הקוד מגדיר את ערך המאפיין ActiveConnection של cat1 (שהוא אובייקט מסוג Catalog), כמסד הנתונים Northwind. פקודת Append נותנת ל-cmd1 את השם AllEmployees ומוסיפה אותו לקטלוג Northwind. לבסוף, MakeAView קוראת ל-ViewAView כדי להדפיס את תוכן התצוגה בחלון **Immediate**.

```
Sub MakeAView()
Dim cnn1 As New ADODB.Connection
Dim cmd1 As New ADODB.Command
Dim cat1 As New ADOX.Catalog
'Open the connection.
    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office\Office\" & _
        "Samples\Northwind.mdb;"
' Create the command representing the view.
' Remember to delete it first.
    Set cmd1.ActiveConnection = cnn1
    cmd1.CommandText = "SELECT FirstName, LastName, " & _
        "Extension FROM Employees"
' Open the catalog.
    Set cat1.ActiveConnection = cnn1
' Create the new view.
    cat1.Views.Append "AllEmployees", cmd1
' Show view.
    ViewAView
End Sub
```

הדפסת תצוגה. השיגרה ViewAView שלהלן מדפיסה את התצוגה AllEmployees. השיגרה פותחת ביצירת חיבור אל מסד הנתונים Northwind ולאחר מכן פותחת את האובייקט rst1 Recordset המבוסס על התצוגה AllEmployees. כשערכת הרשומות פתוחה ומצביעה על מקור הנתונים המתאים, השיגרה עוברת בלולאה על הרשומות

כדי למצוא את השם הארוך ביותר. ערכת הרשומות קטנה יחסית, ולכן אין אנו משלמים מחיר גבוה על ביצוע הלולאה הנוסף. נוח להתבסס על השם הארוך ביותר, מכיון שכך מקבלים נקודת התחלה להדפסת השלוחה לאחר השם. שלב זה מאפשר ליישר את מספרי השלוחות לשמאל ללא תלות באורכי השמות. בטרם מתחילים לעבור על השמות בלולאה נוספת, השיגרה ViewAView חוזרת לתחילת ערכת הרשומות. לאחר מכן, היא עוברת על השמות פעם נוספת, והפעם היא בונה מחרוזת בעזרת המידע שנאסף בלולאה הראשונה.

```
Sub ViewAView()
Dim cnn1 As New ADODB.Connection
Dim rst1 As New ADODB.Recordset
Dim intMaxLength As Integer, Length As Integer

' Open the connection.
    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office\" & _
        "Office\Samples\Northwind.mdb;"

' Find the longest name.
    rst1.Open "AllEmployees", cnn1
    Do Until rst1.EOF
        Length = Len(rst1.Fields("FirstName")) + Len(rst1.Fields("LastName"))
        If Length > intMaxLength Then intMaxLength = Length
        rst1.MoveNext
    Loop

' Print first name, last name, and third field.
    rst1.MoveFirst
    Do Until rst1.EOF
        strFiller = (intMaxLength + 2) - _
            (Len(rst1.Fields("FirstName")) + _
            Len(rst1.Fields("LastName")))
        Debug.Print rst1.Fields("FirstName") & " " & _
        rst1.Fields("LastName") & String(strFiller, " ") & rst1.Fields(2)
        rst1.MoveNext
    Loop
End Sub
```

שינוי תצוגה. השיגרה ChangeAView שלהלן משנה את משפט SQL שעליו מבוססת התצוגה. באפשרותך לנצל גישה פשוטה זו כדי להוסיף שדות חדשים, להחליף שדות קיימים או אפילו לשנות את עיצוב התצוגה בשלמותו, על ידי הוספת רכיבי מיון וסינון. השיגרה משנה את התצוגה כך שתראה את השדה HomePhone במקום את השדה Extension.

```
Sub ChangeAView()
Dim cnn1 As New ADODB.Connection
Dim cat1 As New ADOX.Catalog
Dim cmd1 As New ADODB.Command
' Open the connection.
   cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=C:\Program Files\Microsoft Office\" & _
      "Office\Samples\Northwind.mdb;"
' Open the catalog.
   Set cat1.ActiveConnection = cnn1
' Update the view.
   cmd1.CommandText = "SELECT FirstName, LastName, " & _
      "HomePhone FROM Employees"
   cat1.Views("AllEmployees").Command = cmd1
' Show view.
   ViewAView
End Sub
```

כדי לערוך את משפט SQL שעליו מבוססת התצוגה, יש לפתוח את האובייקט הקשור Command ולערוך את המאפיין CommandText שלו. לשם כך, עליך להפנות את המאפיין ActiveConnection של האובייקט Catalog (ראה cat1 בקוד שלעיל) למסד הנתונים שמכיל את התצוגה שברצונך לשנות – במקרה זה, מסד הנתונים Northwind. הקצה את משפט SQL של התצוגה המבוקשת אל מאפיין CommandText של האובייקט Command החדש. אחר כך הצב את האובייקט Command החדש בתצוגה שברצונך לשנות. ההצבה תשמור אוטומטית את משפט SQL החדש במקום הקודם לו.

מחיקת תצוגה. השיגרה DeleteAView שלפניך מוחקת תצוגה אחת. היא פשוטה ביותר, אך כדאי שתהיה זמינה, אם ברצונך להפעיל את MakeAView יותר מפעם אחת. הסיבה לכך היא ש-ADO אינה מאפשרת לצרף תצוגה אחת על גבי תצוגה אחרת בעלת שם זהה, ולכן עליך למחוק תחילה את התצוגה האחרונה שנוצרה כדי להפעיל מחדש את MakeAView. כדי למחוק תצוגה, הגדר את ActiveConnection של אובייקט Catalog כך שיצביע על מסד נתונים הכולל את התצוגה שברצונך למחוק. לאחר מכן הפעל את השיטה Delete של האוסף Views תוך הפניה לחברים הספציפיים שברצונך למחוק מתוך הסכימה.

```
Sub DeleteAView()
Dim cat1 As New ADOX.Catalog
' Open the catalog.
   cat1.ActiveConnection = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=C:\Program Files\Microsoft Office\Office\" & _
      "Samples\Northwind.mdb;"
' Delete the procedure.
   cat1.Views.Delete ("AllEmployees")
End Sub
```

האובייקט Procedure

שיגרה היא שאילתת החזרת שורות מבוססת-פרמטרים שמחזירה שורות, או שאילתת פעולה שמוסיפה, מוחקת או מעדכנת רשומות. בסעיף זה נעסוק בשאילתות מבוססות-פרמטרים; עיין בפרק 4 לקבלת מידע מפורט יותר על שאילתות, כולל שגרות שמחייבות שימוש בשאילתות פעולה.

קיים דמיון בין שגרות ותצוגות: שתיהן יכולות להשתמש באובייקטי Command. תצוגות מייצגות שאילתות החזרת שורות שאינן מבוססות-פרמטרים, ואילו שגרות מייצגות את שאר סוגי השאילתות. אם תמספר את האובייקטים מסוג Procedure בקטלוג, המספור יכלול את קבוצת כל התצוגות. לעומת זאת, שגרות אינן מופיעות במספור אובייקטים מסוג View. תחביר המספור של שני סוגי השאילתות זהה.

אובייקטים מסוג Procedure כוללים שני מאפיינים קריטיים: Name (שם) ו-Command (פקודה). מאפיין Name הוא שם השיגרה. מאפיין Command מאפשר גישה למאפייני אובייקט Command שעליו מבוססת השיגרה. המאפיין CommandText שימושי במיוחד, מכיון שהוא מגדיר או מחזיר את משפט SQL עבור האובייקט Command. תוכל לנצל מאפיין זה כדי לשנות את משפט SQL באובייקט Command שעליו מבוססת השיגרה, או כדי להציג את משפט SQL של האובייקט. בחינת משפט SQL של האובייקט יכולה ליידע אותך על הפרמטרים שעליך להגדיר כדי להפעיל את השיגרה.

מספור תצוגות ושגרות. השיגרה LinkMyProcs שלהלן מציגה בפועל את האוספים Views (תצוגות) ו-Procedures (שגרות). לולאת For הראשונה ממספרת את חברי האוסף Views בלבד. הלולאה השנייה ממספרת את האוסף Procedures ומציגה את השגרות והתצוגות יחדיו. הלולאה השנייה מציגה גם את משפט SQL של כל שאילתה שהיא ממספרת.

```
Sub ListMyProcs()
Dim cnn1 As New Connection
Dim cat1 As New Catalog
Dim proc1 As Procedure
Dim view1 As View
' Set database connection for catalog.
    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office" & _
        "\Office\Samples\Northwind.mdb;"
    Set cat1.ActiveConnection = cnn1
' Enumerate views -- notice this returns just
' nonparameterized row-returning queries.
    For Each view1 In cat1.Views
        Debug.Print "View name: " & view1.Name
    Next view1
```

```
' Enumerate views -- this returns views and procedures.
     For Each proc1 In cat1.Procedures
        Debug.Print "Procedure name: " & proc1.Name
        If proc1.Name <> "Invoices" Then
           Debug.Print "SQL: " & proc1.Command.CommandText
        End If
     Next proc1
End Sub
```

תצוגות ושגרות מאפשרות להשתמש באובייקטי Command. כל שעליך לעשות הוא לצרף את האובייקט Command לאוסף המתאים. הקוד שלפניך שומר שאילתת פרמטר כשיגרה מאוחסנת. השאילתה משתמשת בטבלה MyTable (שמופיעה בחלק מדוגמאות השגרות שהוצגו בפרק), וניתן להורידה מהתקליטור הנלווה לספר, יחד עם דוגמאות לטיפול בה. התרשימים 2.7 ו-2.8 מציגים את התצוגות **עיצוב** (Design) ו**גיליון נתונים** (DataSheet) של הטבלה MyTable.

```
Sub SaveParameterQuery()
On Error GoTo SavePQTrap
Dim cmd1 As Command
Dim rs1 As Recordset, str1 As String
Dim fldLoop As ADODB.Field
Dim prm1 As ADODB.Parameter, int1 As Integer
Dim cat1 As New ADOX.Catalog
Dim cnn1 As ADODB.Connection

Set cnn1 = CurrentProject.Connection

'Create and define command.
   Set cmd1 = New ADODB.Command

With cmd1
        .ActiveConnection = cnn1
        .CommandText = "Parameters [Lowest] Long;" & _
              "SELECT Column1, Column2, Column3 " & _
              "FROM MyTable " & _
              "WHERE Column1>=[Lowest]"
        .CommandType = adCmdText
   End With

'Open the catalog.
   Set cat1.ActiveConnection = cnn1

' Create the new procedure based on parameter query.
   cat1.Procedures.Append "spLowestRow", cmd1

SavePQExit:
   Exit Sub
```

```
SavePQTrap:
    If Err.Number = -2147217816 Then
'If err.number = -214... query already exists
        deleteProcedure("spLowestRow")
        Resume
    Else
        Debug.Print Err.Number, Err.Description
    End If

End Sub

Sub deleteProcedure(procName as String)
Dim cnn1 As New Connection
Dim cat1 As New adox.Catalog

'Open the catalog.
    Set cnn1 = CurrentProject.Connection
    Set cat1.ActiveConnection = cnn1

'Delete existing procedure.
    cat1.procedures.Delete (procName)

End Sub
```

השיגרה בונה שאילתת פרמטר ושומרת אותה כשיגרה מאוחסנת. השאילתה מבקשת שהערך הנמוך ביותר יופיע בעמודה הראשונה. שים לב כי באפשרותך להגדיר את המאפיינים CommandText ו-CommandType של השאילתה. משפט SQL כולל הצהרה על הפרמטר. לאחר יצירת האובייקט Command של השיגרה המאוחסנת, הפעל את השיטה Append של האוסף Procedures כדי לממש את האובייקט כשיגרה מאוחסנת. אם תפעיל את SaveParameterQuery יותר מפעם אחת, הניסיון השני לשמור את השיגרה המאוחסנת spLowestRow יגרום לשגיאת זמן ריצה. SaveParameterQuery לוכדת שגיאה זו ומוחקת את העותק הישן אם הוא קיים כבר. השיגרה שמוחקת את השיגרה המאוחסנת קולטת ארגומנט עבור שם השיגרה המיועדת למחיקה. השיגרה ParameterQCommand שהוצגה קודם לכן, מבצעת משימה זהה לזו של השיגרה המאוחסנת spLowestRow. ADO מפעילה שגרות מאוחסנות כאובייקטים מהודרים, ולכן הן פועלות מהר יותר מאשר אובייקט Commmand שיש ליצור כל פעם מחדש לשם ביצוע משפט SQL.

יצירה והפעלה של שגרות מאוחסנות. זוג השגרות הבא מכין ומפעיל שיגרה מאוחסנת. הראשונה יוצרת שיגרה מאוחסנת שמחפשת את מספר שלוחת הטלפון של עובדים במסד הנתונים Northwind. השיגרה השנייה מפעילה את השיגרה המאוחסנת.

```
Sub procLookupNumber()
Dim cnn1 As New ADODB.Connection
Dim cmd1 As New ADODB.Command
Dim prm1 As ADODB.Parameter
Dim cat1 As New ADOX.Catalog

' Open the connection.
    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office" & _
        "\Office\Samples\Northwind.mdb;"

' Create the parameterized command.
    Set cmd1.ActiveConnection = cnn1
    cmd1.CommandText = "SELECT FirstName, LastName, Extension " & _
        "FROM Employees WHERE LastName = [LName]"
    Set prm1 = cmd1.CreateParameter("[LName]", adWChar, adParamInput, 20)
    cmd1.Parameters.Append prm1

' Open the catalog.
    Set cat1.ActiveConnection = cnn1

' Create the new procedure based on parameter query.
    cat1.Procedures.Append "spEmployeeExtension", cmd1
End Sub

Sub RunLookUpProc()
Dim cnn1 As New Connection
Dim cat1 As New Catalog
Dim rst1 As New Recordset
Dim cmd1 As New Command
Dim prm1 As Parameter
Dim typedName As String

' Create and assign a connection for the catalog.
    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Program Files\Microsoft Office" & _
        "\Office\Samples\Northwind.mdb;"
    Set cat1.ActiveConnection = cnn1

' Set the Command and parameter object references.
    Set cmd1 = cat1.Procedures("spEmployeeExtension").Command
    Set prm1 = cmd1.CreateParameter("[LName]", adWChar, adParamInput, 20)
    cmd1.Parameters.Append prm1
```

```
' Gather the parameter value from the user and assign it.
    typedName = InputBox("Last name for extension?", _
        "Programming Microsoft Access 2000")
    prm1.Value = typedName

'If prm1.Value <> "" Then
'Execute the parameter query and show first match.
        cmd1.Execute
        rst1.Open cmd1
        MsgBox "The extension for " & rst1.Fields(0) & _
            " " & rst1("LastName") & " is " & rst1.Fields(2), _
            vbInformation, "Programming Microsoft Access 2000"
    End If
End Sub
```

השיגרה ProcLookupNumber יוצרת שאילתת פרמטר כשיגרה מאוחסנת. שאילתה זו מחפשת אחר מספר שלוחת טלפון של עובד, לפי שם המשפחה שלו. השיגרה מגדירה את האובייקט Command והפרמטר הקשור אליו [LName]. הקבוע AdWChar מגדיר את הפרמטר כמחרוזת טקסט ברוחב קבוע. הקבוע adParamInput מגדיר את הפרמטר כקלט בלבד, והמספר הנגרר מציין שהפרמטר יכול להכיל עד 20 תווים. לאחר הגדרת האובייקט Command והפרמטר הקשור אליו, השיגרה מגדירה את חיבור הקטלוג אל מסד הנתונים Northwind ומצרפת את האובייקט Command לאוסף Procedures תחת השם spEmployeeExtension.

אם תפעיל את procLookupNumber יותר מפעם אחת, היא תיכשל בעת ניסיון לכתוב עותק חדש של spEmployeeExtension על גבי זה הקיים. הוסף לשיגרה קוד ללכידת שגיאות לטיפול במצב מעין זה. לאחר מכן צור שגרת מחיקה. יהיה עליך לערוך את שגרת המחיקה שבדוגמה הקודמת, מכיון שהיא מוחקת שגרות בפרויקט הנוכחי, בעוד שבדוגמה זו עליה למחוק שיגרה במסד הנתונים Northwind.

השיגרה RunLookUpProc מפעילה את השיגרה המאוחסנת שנוצרה על ידי השיגרה procLookupNumber. שיגרה קצרה זו מבצעת ארבע מטלות :

- יצירת חיבור אל מסד הנתונים באמצעות השיגרה המאוחסנת.
- הצבת משתני אובייקט שמצביעים על השיגרה המאוחסנת והפרמטרים שלה. הקוד מנצל אובייקט פקודה כדי להפנות אליה.
- הצגת תיבת דו-שיח בה המשתמש חייב לציין את שם המשפחה של העובד. הקוד מציב את ערך ההחזרה מתוך תיבת הדו-שיח בפרמטר הפקודה.
- ביצוע הפקודה ופתיחת ערכת רשומות המבוססת על קבוצת ההחזרה של הפקודה. השיגרה מציגה את השורה הראשונה מקבוצת ההחזרה בתיבת דו-שיח. הקוד משתמש בשתי מערכות כללי מתן השמות כדי לבחור חבר מתוך האוסף Fields בערכת רשומות. תוכל לבחור את המיקום הסידורי או את המאפיין Name שלו.

3

עיצוב טבלאות

עסקים רבים המשתמשים במסדי נתונים, אינם מייחסים את החשיבות הראויה לעיצוב טבלאות. במקום זאת הם נוטים להתמקד בעיצוב טפסים ודוחות, אלמנטים הגלויים למשתמש. חשוב להדגיש כי העיצוב והתוכן של טבלאות וקשרי הגומלין ביניהן, יכולים להשפיע משמעותית על השירותים שניתן לספק באמצעות פתרונות מסד נתונים מותאמים אישית. עיצוב טבלאות יכול להשפיע על ביצועי מסדי הנתונים האלה וגם על קלות הוספת שירותים חדשים למערכת.

פרק זה פותח בסקירה קצרה של שיקולים כלליים בעיצוב טבלאות, כגון פיצול המידע בקבוצות לוגיות כדי שניתן יהיה לאחסנו בטבלאות שונות. אחר דן הפרק ביצירת טבלאות באמצעות Access 2000, בצורה ידנית ובעזרת האשף של Access. נלמד גם אודות אינדקסים, מפתחות ראשיים ועל אופן יצירת קשרי גומלין בין טבלאות.

הפרק מלמד כיצד ליצור טבלאות באופן דינמי ולמלא אותן בנתונים, יכולת שמאפשרת ליצור פתרונות מסדי נתונים רבי-עוצמה ועתירי רכיבים. התקליטור המצורף מכיל דוגמאות רבות של אובייקטי נתונים ADO - ActiveX (ActiveX Data Objects) - שיוצרים טבלאות Access באופן דינמי וממלאים אותן בנתוני Access. דוגמאות אלו מבוססות על המבוא ל-ADO שהוצג בפרק 2. קבוצת דוגמאות אחת אף מציגה כיצד לשלב את ADO ואת האובייקט הידידותי DoCmd כדי לתקשר עם מקורות נתונים ISAM (Indexed Sequential Access Method) ו- ODBC (קישוריות מסד נתונים פתוחה).

טבלאות ומסדי נתונים יחסיים

טבלאות נמנות עם יחידות המבנה הבסיסיות ביותר של מסד נתונים יחסי. דוגמת מסד הנתונים Northwind המשווקת עם Access, היא דוגמה למסד נתונים המנהל עסק דמיוני. מסד נתונים זה כולל טבלאות אחדות העוקבות אחר מידע חשוב. הטבלה Customers עוקבת אחר מידע על לקוחות, כגון שם חברה ושם איש קשר. הטבלה Products עוקבת אחר נתונים הקשורים למוצר, כגון שם המוצר, מספר יחידות במלאי ומחיר היחידה.

כל טבלה במסד נתונים חייבת להכיל מידע המתאים ליישות מסוג מוגדר אחד בלבד. לדוגמה, בית-ספר טיפוסי כולל בין היתר, תלמידים וכיתות. תלמידים רשומים בכיתות; מורים מלמדים בכיתות. מסד הנתונים של בית הספר עשוי להכיל טבלת תלמידים (Students), טבלת כיתות (Classes) וטבלת מורים (Teachers). הטבלה **מורים** צריכה להכיל מידע על מורים, אך לא צריך להיות בה מידע על תלמידים או כיתות. באופן דומה, הטבלה **כיתות** אינה אמורה להכיל פרטי מורים, כגון כתובות, מספרי תעודת זהות ומספרי טלפון.

טבלאות מסד נתונים דומות במבנה שלהן לגליונות אלקטרוניים. **שורות** (Rows) או **רשומות** (Records) בטבלה מייצגות מופעים ייחודיים של היישויות המאוחסנות בטבלה. לדוגמה, הטבלה Customers כוללת שורה אחת לכל לקוח, ואילו הטבלה Products כוללת שורה אחת עבור כל מוצר (אין חשיבות כלשהי לסדר השורות; תוכל לסדר אותן כרצונך מבלי לשנות את משמעות המידע שבטבלה).

כל **שדה** (Field) בטבלה מכיל מידע מסוג מוגדר. לדוגמה, לכל אחד מהלקוחות במסד הנתונים Northwind יש שם, כתובת, מספר טלפון וכן הלאה - מידע זה מאוחסן בשדות הטבלה Customers.

טבלאות מסד נתונים רבות כוללות שדה אחד או יותר שמזהה באופן ייחודי כל שורה בטבלה. זיהוי ייחודי זה נקרא **מפתח ראשי** (Primary Key). המפתח הראשי של הטבלה Customers שבמסד הנתונים Northwind הוא השדה CustomerID (מזהה לקוח).

נרמול

נרמול (normalization) הוא החלה של קבוצת כללי עיצוב על טבלאות מסד נתונים. לנרמול ארבעה יתרונות עיקריים:

- **ביטול מידע מיותר**. מסדי נתונים רבים שלא עברו נרמול מחייבים להכניס אותם נתונים של אנשי קשר בצורות רבות. ביטול היתירות מצמצם סיכויי קליטה של נתונים שגויים, דבר שעלול לשבש את מסד הנתונים. הנרמול יכול גם לפשט את תחזוקת מסד הנתונים, כיון שקל יותר למחוק או לעדכן ערך המאוחסן במקום אחד בלבד.

- **צמצום גודל מסד הנתונים.** כל סוג מידע מאוחסן במקום אחד בלבד, ולכן מסד הנתונים אינו צריך לשמור עותקים רבים של פריטי מידע זהים. הנרמול אף מצמצם את מספר העמודות בטבלה, מה שמקטין את מסד הנתונים כולו.
- **פישוט תהליכי חיפוש.** מקצועני מסדי נתונים המתמצאים בכללי הנרמול, יידעו כיצד לנווט בטבלאות מסד הנתונים לשם איתור מידע. משתמשי מסדי נתונים מזדמנים יצליחו להבין טוב יותר את עיצוב הטבלה, מכיון שכל טבלה מייצגת יישות יחידה שכל מאפייניה הן עמודות בטבלה זו.
- **פישוט ביצוע שאילתות.** עמודת טבלה מאחסנת סוג נתונים יחיד, כגון שם פרטי או שם משפחה, אך לא את שני סוגי השם. אחסון שם המשפחה בעמודה נפרדת מאפשר להפיק בקלות רשימה של כל שורות מסד הנתונים המכילות שם משפחה זהה. במסד נתונים לא מנורמל שמאחסן שמות פרטיים ושמות משפחה באותה עמודה, יש לחלץ תחילה את שם המשפחה לפני ביצוע שאילתה המטפלת בשמות.

קיימים שלושה כללי נרמול מקובלים: **First Normal Form** (כלל הנרמול הראשון), **Second Normal Form** (כלל הנרמול השני) ו- **Third Normal Form** (כלל הנרמול השלישי). קיימים גם כללים מיוחדים רבים נוספים.

First Normal Form - כלל הנרמול הראשון

כלל הנרמול הראשון קובע כי כל שדה בטבלה חייב להכיל פריט יחיד שאינו ניתן לחלוקה. הטבלה Order Details של מסד הנתונים Northwind ממחישה חוק זה. תרשים 3.1 מציג קטע מהטבלה ובו שלוש הזמנות. כל עמודה בטבלה מכילה סוג נתונים יחיד. העמודה השנייה מכילה שמות מוצרים, אך לא שמות ומחירים גם יחד. בנוסף, כל שורה בטבלה כוללת מופע אחד בלבד של מוצר.

Order Details : Table

Order ID	Product	Unit Price	Quantity	Discount
10248	Queso Cabrales	$14.00	12	0%
10248	Singaporean Hokkien Fried Mee	$9.80	10	0%
10248	Mozzarella di Giovanni	$34.80	5	0%
10249	Tofu	$18.60	9	0%
10249	Manjimup Dried Apples	$42.40	40	0%
10250	Jack's New England Clam Chowder	$7.70	10	0%
10250	Manjimup Dried Apples	$42.40	35	15%
10250	Louisiana Fiery Hot Pepper Sauce	$16.80	15	15%

Record: 1 of 2155

תרשים 3.1: קטע מהטבלה Order Details הממחיש את כלל הנרמול הראשון. האלמנטים בכל עמודה (או שדה) נמנים עם סוג נתונים יחיד; האלמנטים שבעמודות השורה אינם ניתנים לפירוק

טבלאות מסד נתונים שאינן עומדות בדרישות כלל הנרמול הראשון, מתאפיינות בדרך כלל בשתי תכונות: ראשית, הן מאחסנות יותר מפריט אחד באותו שדה (שים לב שהטבלה בתרשים 3.1 מכילה פריט בודד בכל שדה). דוגמה לאי-עמידה בדרישות הכלל הראשון היא טבלה ששדה השם שלה מכיל שם פרטי ושם משפחה. צירוף סוגי

מידע בשדה משותף מסבך את תהליך שליפת המידע. הפרה שכיחה נוספת של כלל הנרמול הראשון היא טבלה שמכילה שדה לכל מוצר בהזמנה, דבר שמחייב ליצור טבלה בעלת שדות רבים של מחיר, כמות והנחה, ובהמשך גורם להפרת הכלל הראשון. עיצוב כזה "מנפח" את הטבלה, וחלק גדול מערכי עמודות אחדות יהיו ריקים.

טבלאות העומדות בדרישות כלל הנרמול הראשון מבוססות בדרך כלל על יישויות לוגיות שמאפשרות לאתר מידע בקלות. לעיתים קרובות יש להן מפתח ראשי שמבטיח כי כל שורה בטבלה היא ייחודית. כזכור, טבלאות מסדי נתונים מייצגות מופעי יישויות. בטבלה שבתרשים 3.1, כל שורה מייצגת פריט בהזמנת לקוח. המפתח הראשי של הטבלה הוא מפתח מורכב (Compound Key) שמבוסס על השדות OrderID ו-ProductID, מכיון שניתן לזהות כל פריט בהזמנת הלקוח באמצעות צירוף ייחודי של מספר הזמנת הלקוח ומזהה המוצר.

Second Normal Form – כלל הנרמול השני

כלל הנרמול השני מציין קשרי גומלין נדרשים בין העמודות בשורה. כדי לעמוד בדרישות הכלל השני, טבלה חייבת לעמוד קודם כל בתנאי כלל הנרמול הראשון. בנוסף, כל הערכים בשורה חייבים להכיל מידע על היישות הספציפית שמיוצגת באמצעות השורה. דרישה נוספת: אסור שתהיה תלות הדדית בין שני שדות כלשהם. אחת ההפרות הנפוצות של כלל זה היא הצבת שדות לשתי יישויות ייחודיות בשורה אחת.

הטבלה Order Details של Northwind עומדת בדרישות חוק הנרמול השני. שדה הכמות מתאים בצורה ברורה לכל מוצר. מחיר היחידה וההנחה יכולים להשתנות בצורה עצמאית לכל פריט, כאשר הצבת מחיר יחידה והנחה בטבלה מאפשרת ליישום לעקוב בקלות אחר פריטים אלה לכל הזמנה.

במסד הנתונים Northwind קיים שדה מחיר יחידה הן בטבלה Order Details והן בטבלה Products. זה נראה כמו הפרה של כלל הנרמול השני, אולם עיצוב זה מאפשר למנהל לעדכן מחירי פריטים מבלי להשפיע על מחירי פריטים שכבר הוזמנו. ההזמנות יכולות לנצל את המחיר וההנחה שבטבלה Products בתור הנחיה ולאו דווקא ככלל ברזל לכל העסקאות.

Third Normal Form – כלל הנרמול השלישי

כלל הנירמול השלישי קובע כי כל השדות בכל שורה חייבים להיות ייחודיים ובלתי תלויים זה בזה. לדוגמה, החוק מאפשר רק שדה תאריך אחד בשורה. אם הטבלה Orders מכילה שדה תאריך הזמנה, אסור שתכיל שדות נוספים של יום, שבוע או חודש של ההזמנה (היישום יכול לחלץ בעצמו את החודש מתוך תאריך ההזמנה). הטבלה Orders מכילה שדות אחדים שמכילים ערכי שעה/תאריך (Date/Time), אך אלה מתייחסים למועד ההזמנה המקורי, התאריך בו הגיעה דרישת הלקוח, יום האספקה וכן הלאה.

עמידה בכללי הנרמול

נבון יהיה למלא את דרישות הכללים, אך בנסיבות מסוימות ייתכן שתרצה לעשות זאת בצורה סלקטיבית - למשל, אם ברצונך לצמצם את מספר הטבלאות. השדות מיקוד, קוד אזור וקוד מדינה (בארה״ב) יכולים להתקיים בטבלה נפרדת. אך אם ידוע לך המיקוד של כתובת מסוימת, ידועים לך גם העיר, האזור, או המדינה, ולכן אינך צריך לשמור את המיקוד ושם העיר באותה טבלה.

בפרויקטים מסוימים של מסדי נתונים קטנים ובינוניים יצירת טבלאות השומרות מידע על מיקוד, שם עיר, אזור ומדינה עלולה להיות בלתי מעשית. גם בפרויקט מסד נתונים גדול מאוד, כשנדרשים פרטי אנשי קשר, תכנון מסוג זה מחייב לצרף את הטבלה הראשית Contacts עם טבלאות רבות בכל מקרה הדורש מידע אנשי קשר. הדבר עלול לפגוע קשות בביצועי המערכת, במיוחד אם צירוף הטבלאות נדרש פעמים רבות.

קשרי גומלין בין טבלאות

בעת יצירת טבלאות, עליך להביא בחשבון גם את קשרי הגומלין ביניהן. קשרים אלה מעניקים למסד נתונים **יחסי** (relational) את רוב עוצמתו. קיימים שלושה סוגי קשרי גומלין: **יחיד-ליחיד** (One-to-One), **יחיד-לרבים** (One-to-Many) **ורבים-לרבים** (Many-to-Many).

קשרי גומלין מסוג יחיד-ליחיד

במערכת קשרי גומלין מסוג יחיד-ליחיד, כל רשומה בטבלה אחת מתאימה לרשומה יחידה בטבלה אחרת. סוג קשר זה אינו נפוץ ביותר, אך ביכולתו להציע מספר יתרונות. ראשית, תוכל להציב שדות משתי הטבלאות בטבלה משולבת. אחת הסיבות לשימוש בשתי טבלאות היא שכל שדה הוא מאפיין של יישות נפרדת, כגון נהגים ומשאיות. כל נהג יכול לנהוג משאית אחת בלבד בזמן נתון, אך שדות הטבלאות של הנהגים והמשאיות מתייחסים ליישויות שונות.

קשרי גומלין מסוג יחיד-ליחיד יכולים גם לצמצם את הזמן הדרוש לפתיחת טבלה גדולה על ידי הצבת חלק מעמודות הטבלה בטבלה נפרדת. גישה זו הגיונית במיוחד כשטבלה מכילה שדות אחדים שהשימוש בהם אינו תכוף. לבסוף, קשרי גומלין מסוג יחיד-ליחיד יכולים לתמוך באבטחה. Access מיישם אבטחה ברמת-משתמש ברמת הטבלה. כך, אם קבוצת-משנה של שדות בטבלה מחייבת הגנה באמצעות אבטחה, הצבת שדות אלה בטבלה נפרדת מאפשרת ליישום להגביל את הגישה לשדות מסוימים. היישום יכול לקשר את הטבלה המוגבלת בחזרה לטבלה הראשית באמצעות קשרי גומלין מסוג יחיד-ליחיד, ועל ידי כך לאפשר למי שמחזיק בהרשאות מתאימות לערוך, למחוק ולהוסיף רשומות חדשות לשדות אלה.

קשרי גומלין מסוג יחיד לרבים

נפוץ יותר הוא הקשר יחיד-לרבים, שבו שורה מתוך טבלה אחת מתאימה לשורה אחת או יותר מהטבלה השנייה. סוג קשר זה יכול ליצור גם בסיס לקשרי גומלין מסוג רבים-לרבים. הטבלאות Customers (לקוחות) ו-Orders (הזמנות) של מסד הנתונים Northwind מקיימות קשרי גומלין מסוג יחיד-לרבים. ללקוח כלשהו יכולה להיות הזמנה אחת או יותר, אך כל הזמנה שייכת רק ללקוח אחד. תרשים 3.2 מציג את תצוגת **עיצוב** (Design) של הטבלה Orders. המפתח הראשי הוא OrderID. כל אחד מערכי OrderID יכול להופיע בשורה אחת בלבד. הטבלה מכילה גם שדה שנקרא CustomerID, שמקושר לטבלה Customers. השדה CustomerID הוא המפתח הראשי של הטבלה Customers. שדה המשמש כקישור לטבלה אחרת, נקרא **מפתח זר** (Foreign Key). ערכי CustomerID יכולים להופיע פעמים רבות בטבלה Orders. לאמיתו של דבר, קוד זיהוי הלקוח מופיע פעם אחת לכל הזמנה שהלקוח מבצע. מפתח זר זה מקשר את צד ה"רבים" של מערכת קשרי הגומלין חזרה לצד ה"יחיד".

Orders : Table

	Field Name	Data Type	Description
🔑	OrderID	AutoNumber	Unique order number.
▶	CustomerID	Text	Same entry as in Customers table.
	EmployeeID	Number	Same entry as in Employees table.
	OrderDate	Date/Time	
	RequiredDate	Date/Time	
	ShippedDate	Date/Time	
	ShipVia	Number	Same as Shipper ID in Shippers table.
	Freight	Currency	
	ShipName	Text	Name of person or company to receive the shipment.
	ShipAddress	Text	Street address only -- no post-office box allowed.

Field Properties

General | Lookup

Field Size	5
Format	
Input Mask	
Caption	Customer
Default Value	
Validation Rule	
Validation Text	
Required	No
Allow Zero Length	No
Indexed	Yes (Duplicates OK)
Unicode Compression	Yes

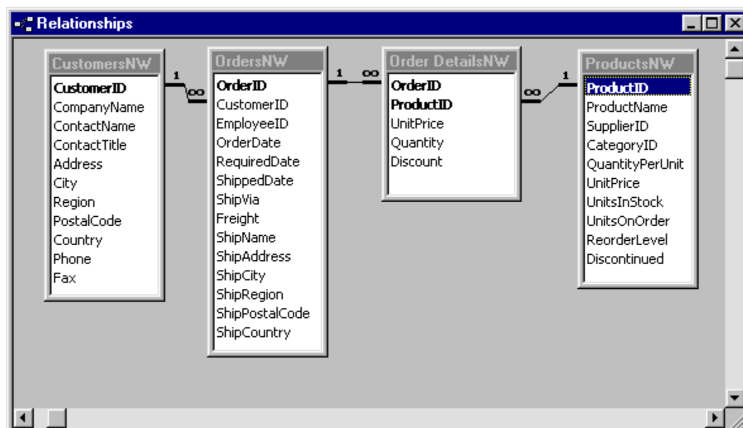
A field name can be up to 64 characters long, including spaces. Press F1 for help on field names.

תרשים 3.2: תצוגת **עיצוב** של הטבלה Orders; השדה CustomerID משמש כמפתח זר בקשרי הגומלין מסוג יחיד-לרבים בין הטבלה Orders לטבלה Customers

קשרי גומלין מסוג רבים-לרבים

קשרי גומלין מסוג רבים-לרבים קיימים בצורה עקיפה בלבד ומבוססים על שתי מערכות קשרי גומלין מסוג יחיד-לרבים. הטבלאות Orders ו-Products מקיימות ביניהן קשרי גומלין מסוג יחיד-לרבים. הזמנה כלשהי יכולה להכיל מוצרים רבים. באופן דומה, מוצר יחיד יכול להופיע בהזמנות רבות ושונות.

ממסד הנתונים Northwind יובאו ארבע טבלאות לדוגמת מסד הנתונים של פרק זה: CustomersNW, ‏Orders DetailsNW, ‏DetailsNW ו-ProductsNW. תרשים 3.3 מציג בחלון **קשרי גומלין** (Relationships) את קשרי הגומלין בין הטבלאות. הטבלה CustomersNW מקיימת קשרי גומלין מסוג יחיד-לרבים עם הטבלה OrdersNW, הטבלה OrdersNW מקיימת קשרי גומלין מסוג יחיד-לרבים עם הטבלה Order DetailsNW והטבלה ProductsNW מקיימת קשרי גומלין מסוג יחיד-לרבים עם הטבלה Order DetailsNW. הטבלה Order DetailsNW משמשת כ"תיבת הסתעפויות" שמקשרת את הטבלאות OrdersNW ו-ProductsNW בקשרי גומלין מסוג רבים-לרבים.



תרשים 3.3: החלון **קשרי גומלין** מציג את קשרי הגומלין מסוג רבים-לרבים בין הטבלאות OrdersNW ו-ProductsNW

קשרי גומלין מסוג רבים-לרבים מעוצבים במבנה "תיבת הסתעפויות", כשטבלה שלישית, תיבת ההסתעפויות, משמשת כקישור בין שתי הטבלאות האחרות במערכת קשרי הגומלין. עיצוב זה מסייע לשמור על יעילות מסד הנתונים, כיון שתיבת ההסתעפויות יכולה להכיל מספר שורות גדול, אך מספר עמודות קטן. בדומה לכל הטבלאות שעברו נרמול, עליך להגביל את העמודות לאלו שמאפיינות את היישויות. כלל זה חשוב במיוחד לגבי טבלאות מסוג תיבת הסתעפויות, מכיון שהן עלולות להכיל מספר שורות גדול יחסית.

שלמות הקשרים

אפשר להקנות חוסן למסד הנתונים באמצעות **שלמות קשרים** (Referential integrity) **ועדכונים ומחיקות על פי היררכית קשרים** (Cascading updates and deletes). כללי שלמות הקשרים מבטיחים את חוקיות קשרי הגומלין בין טבלאות. כללים אלה גם מונעים שינוי בשוגג של נתונים קשורים (לדוגמה, לא כדאי למחוק לקוח, אם יש לו הזמנה שלא שולמה או מספר הזמנות כאלו).

שלמות הקשרים אינה מאפשרת הוספת שורה לצד ה״רבים״ בקשרי הגומלין באמצעות ערך מפתח זר שאינו תואם לערכים שבצד ה״יחיד״ של הקשר. אך אפשר להכניס ערך מפתח זר חסר או ריק (Null) שאינו תואם מפתח ראשי כלשהו בצד ה״יחיד״ של קשרי הגומלין.

שלמות הקשרים מסייעת גם להימנע ממצב של רשומות יתומות - רשומות שנמצאות בצד ה״רבים״ של קשרי הגומלין, שאין להן בנות-זוג בצד ה״יחיד״. הדבר מתאפשר הודות לחסימת אפשרות מחיקת רשומות בצד ה״יחיד״, אם יש להן עדיין בנות-זוג בצד ה״רבים״ (לדוגמה, מן הסתם לא יעלה על דעתך להסיר את המכירות הקשורות לעובד מסוים רק משום שהעובד עזב את החברה). רשומות יתומות מנפחות את מסד הנתונים ברשומות מיותרות ולא שימושיות; במקרים קיצוניים יותר, הן עלולות לערער את חוקיות התוצאות המושגות ממסד הנתונים.

לעיתים תרצה שהשינויים יתבצעו בשני צדי קשר הגומלין. במצבים אלה תכונות המחיקה והעדכון על פי היררכית קשרים של שלמות הקשרים עשויים להיות שימושיים. תכונות אלו אינן מופעלות אוטומטית; עליך להפעילן במפורש. אם תפעיל מחיקה על פי היררכית קשרים, מחיקת רשומה כלשהי בצד ה״יחיד״ של קשר הגומלין תסיר את כל הרשומות התואמות בצד ה״רבים״. במקרה של שינויים במפתח ראשי עדכונים על פי היררכית קשרים פועלים בצורה דומה. אם תשנה ערך מפתח ראשי בצד ה״יחיד״ של קשר הגומלין, הוא יתבטא גם בצד ה״רבים״. כל ערכי המפתח הראשי התואמים מתעדכנים אוטומטית.

יצירת טבלאות באמצעות אשפים

אשפי Access מפשטים מטלות רבות, כגון יצירה ותחזוקה של טבלאות. ל-Access מעל 20 אשפי מסד נתונים, למשל: Contact Management, Time and Billing, Service Call Management ו- Collection Management Video. כל אלה יוצרים יישומים שלמים, כולל עיצוב הטבלאות. אשף הטבלאות יוצר טבלאות יחידות ואף מוסיף לכל אחת מהן מפתחות ראשיים ומקשר ביניהן. האשף **בונה שדות** (Field Builder) מסייע לתחזק טבלאות על ידי פישוט הוספת השדות.

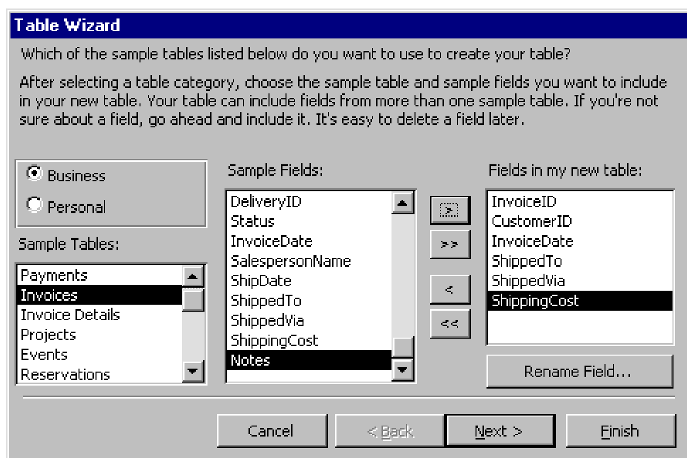
אשפי מסד הנתונים

בעת הפעלת Access 2000 תוכל להציג את אשפי מסד הנתונים, על ידי בחירה באפשרות **אשפי מסד נתונים, דפים ופרויקטים של Access** (Access Database Wizards, Pages And Projects) בתיבת הדו-שיח **Microsoft Access** ולחיצה על **אישור**. בהופעת תיבת הדו-שיח **חדש** (New), בחר בכרטיסיה **מסדי נתונים** (Databases) ואחר כך לחץ לחיצה כפולה על סמל האשף שברצונך להפעיל (לחילופין, ניתן לפתוח את תיבת הדו-שיח מתפריט **קובץ** (File), האפשרות **חדש**).

מסד נתונים שנוצר על ידי אשף מושלם מבחינה תפקודית למשתמש לא מנוסה, מכיון שהוא כולל טפסים, טבלאות, דוחות ואף נתונים לדוגמה. תוכל לבחון את עיצוב מסד הנתונים ולהשתמש בו כמודל ליישומים מותאמים אישית. מפתחים מנוסים יותר יכולים לערוך את העיצוב הבסיסי על ידי הוספת טבלאות, שאילתות, טפסים, דוחות ומודולים.

אשף הטבלאות

אשף הטבלאות מתמקד בלעדית בעיצוב טבלה. תוכל לפתוח אותו מתוך הפריט **טבלאות** (Tables) באוסף **אובייקטים** (Objects) שבחלון **מסדי נתונים** (Database). לחץ לחיצה כפולה על הסמל **יצירת טבלה באמצעות אשף** (Create Table By Using Wizard) כדי לפתוח את תיבת הדו-שיח ההתחלתית **אשף הטבלאות** (Table Wizard). אשף זה מציע מיגוון עיצובים מוכנים מראש בסגנון עסקי או אישי. עיצובי הטבלאות העסקיים כוללים בין היתר את **אנשי קשר** (Contacts), **לקוחות** (Customers), **עובדים** (Employees), **מוצרים** (Products), **הזמנות** (Orders), **חשבוניות** (Invoices) **ואירועים** (Events). עיצובי הטבלאות האישיים עונים על כל הצרכים ויכולים לשרת גם מיגוון צרכים של עסקים קטנים. בין העיצובים המוצעים: **מתכונים** (Recipes), **צמחים** (Plants), **יומן תרגילים** (Exercise Log), **ספרים** (Books), **אוסף וידאו** (Video Collection) **והקלטות** (Recordings). האשף אינו ממלא את הטבלה בנתונים ראשוניים, אך הוא מספק קישורים לטבלאות אחרות שכבר כלולות בעיצוב מסד הנתונים. הוא אפילו יבנה קשרי גומלין אלה בעצמו.

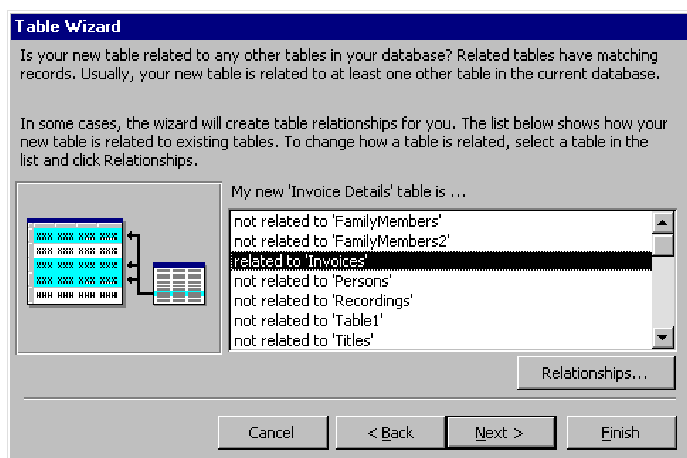


תרשים 3.4: יצירת הטבלה **חשבוניות** בתיבת הדו-שיח הפותחת של **אשף הטבלאות**

לאחר בחירת עיצוב טבלה מוכן מראש, תוכל למלא את הגירסה המותאמת אישית שיצרת בשדות מתוך תיבת רשימה. תיבת הדו-שיח ההתחלתית **אשף הטבלאות** כוללת קבוצת לחצנים להעברת שמות שדה בין תיבת הרשימה **שדות הדוגמה** (Sample Fields) לתיבת הרשימה **שדות בטבלה החדשה** (Fields in my new table). תרשים 3.4 מציג שלב ביצירת טבלת חשבוניות. שדות מדגמיים אחדים הועברו לתיבת הרשימה **שדות**

בטבלה החדשה. בשלב זה, תוכל ללחוץ שלוש פעמים על לחצן **הבא** (Next) כדי ליצור את הטבלה **חשבוניות** (Invoices). הגדרות ברירת המחדל יוצרות מפתח ראשי, מקשרות בצורה טבעית את הטבלה החדשה לטבלה כלשהי במסד הנתונים, ומאפשרות להכניס נתונים. לאחר קבלת כל ברירות המחדל ולחיצה על לחצן **סיום** (Finish), הטבלה תיפתח בתצוגת **גיליון נתונים** (Datasheet) ותאפשר למלא אותה בנתונים. בעת המעבר בין תיבות הדו-שיח של האשף תוכל לדרוס כל ברירת מחדל לא רצויה.

אם התחלת זה עתה את צעדיך הראשונים בעבודה עם מסדי נתונים יחסיים, זיהוי הקישורים האוטומטי לטבלאות נוספות שימושי במיוחד. תרשים 3.5 מציג את תיבת הדו-שיח השלישית של יצירת הטבלה **פרטי חשבונית** (Invoice Details). הטבלה יכולה לכלול את השדה InvoiceID הפועל כמפתח זר של הטבלה **חשבוניות** (Invoices). אם האשף מזהה את הטבלה **חשבוניות**, הוא מקשר אוטומטית בין שתי הטבלאות. תוכל לדרוס את הקישור, או ליצור קישורים שונים מאלה שמציע האשף, אך אם אין לך ניסיון מוטב שתקבל את ברירות המחדל המוצעות.



תרשים 3.5: תיבת דו-שיח של **אשף הטבלאות** המציגה זיהוי של קישור אוטומטי בין הטבלה **פרטי חשבונית** והטבלה **חשבוניות** שנוצרה לפניה

בונה השדות

לאחר שיצרת טבלה באמצעות אשף, יתכן שתרצה להוסיף לה שדה נוסף. המשימה עלולה להתגלות כקשה יותר מאשר היא נראית מלכתחילה, במיוחד אם השדה צריך להיות מקושר אל שדות נוספים בטבלאות אחרות. הדבר נובע מכך שקל יותר ליצור שדה חדש בעל סוג נתונים שונה מאשר ליצור שדה בטבלה אחרת. כשמנסים לקשר בין שדות שנמצאים בטבלאות שונות, ששמותיהם דומים וסוגי נתוניהם שונים, הפעולה תיכשל מבלי שתקבל הסבר כלשהו. פתרון אפשרי לבעיית ההוספה של שדות חדשים לטבלה הוא **בונה שדות** (Field Builder).

פותחים את **בונה שדות** על ידי לחיצה ימנית על שורה ריקה בתצוגת **עיצוב** (Design) של טבלה ובחירה ב**בניה** (Build) מתוך תפריט הקיצור. תיבת הדו-שיח **בונה שדות** כוללת שתי תיבות רשימה: אחת לבחירת סוג טבלה, כגון **חשבוניות** או **פרטי חשבונית**, ותיבה נוספת לבחירת שדה מתוך סוג הטבלה שנבחר. בעת הארת סוגי טבלה שונים, משתנים הערכים בתיבת הרשימה **שדות דוגמה** (Sample Fields). חפש את השדה שברצונך להוסיף ולחץ על **אישור** (OK) כדי להוסיפו לטבלה. פעולה זו מבטיחה שהשדות החדשים שהוספת יכילו סוגי נתונים תואמים לשדות שהוספו באמצעות אשף הטבלאות.

יצירת טבלה באופן ידני

ליצירת טבלה באופן ידני, לוחצים לחיצה כפולה על הסמל **יצירת טבלה בתצוגת עיצוב** (Create Table In Design View), כדי לפתוח חלון **טבלה** (Table) ריק בתצוגת **עיצוב** (Design). מכאן ניתן להוסיף שדות, מפתח ראשי ואינדקס (מפתחות ראשיים הם אינדקסים בעלי הגדרות מאפיינים מיוחדות).

להוספת שדה, הקלד את שמו בעמודה הריקה **שם שדה** (Field Name) שבחלון תצוגת **עיצוב**. שמות שדות עומדים בכללי מתן השם הרגילים של Visual Basic for Applications (VBA). אורכם המקסימלי יכול להגיע ל-64 תווים – אותיות, מספרים, רווחים ותווים מיוחדים, למעט התווים נקודה, סימן קריאה, סוגריים מרובעים סימון הטעמה מסוג grave (`). כמו כן, אסור לפתוח שם שדה בתו רווח או בתו בקרה (ערך ASCII בטווח 0-31). תווי רווח יכולים להופיע באמצע שמות שדה, אך בביטויים ובשאילתות יש להציבם בין סוגריים רגילים.

סוגי נתונים

העמודה **סוג נתונים** מאפשרת לציין את סוג הנתונים של השדה. תיבת רשימה נפתחת מציעה 10 אפשרויות: **טקסט** (Text), **תזכיר** (Memo), **מספר** (Number), **תאריך/שעה** (Date/Time), **מטבע** (Currency), **מספור אוטומטי** (AutoNumber), **כן/לא** (Yes/No), **אובייקט OLE** (OLE Object), **היפר-קישור** (Hyperlink) **ואשף בדיקת מידע** (Lookup Wizard) (סוגי נתונים משמשים בדרך כלל לזיהוי המידע שבשדה. שדה **טקסט** מכיל נתונים מסוג **טקסט**, שדה **מספור אוטומטי** מכיל נתונים מסוג **מספור אוטומטי** וכן הלאה). חלק גדול מסוגי הנתונים כוללים אפשרויות שונות להגדרה מדויקת יותר של סוג הנתונים. לדוגמה, סוג הנתונים **מספר** כולל שבעה סוגי משנה. באפשרותך להגדיר אחד מהם על ידי בחירת **מספר** בתור סוג הנתונים ולאחר מכן לבחור באפשרות הרצויה מתוך תיבת הרשימה הנפתחת **גודל שדה** (Field Size) בכרטיסיה **כללי** (General) בחלקו התחתון של החלון **טבלה**.

שדות מספור אוטומטי

שדות מסוג **מספור אוטומטי** (AutoNumber) משמשים לעיתים קרובות בתור מפתח ראשי של טבלה. Access מציב בעצמו ערך חדש בשדה בעת הוספת רשומה לטבלה. שדה זה אינו בר-עדכון בצורה ידנית, ולכן ערכיו אידיאליים לסימון ייחודי של שורה בטבלה.

Access מגדיר בעצמו את הערך של שדה מסוג **מספור אוטומטי**. כדי לגרום לשדה כזה לגדול בהפרש קבוע בצורה סדרתית, בחר באפשרות **תוספת קבועה** (Increment) (שהיא ערך ברירת המחדל) מתוך תיבת הרשימה **ערכים חדשים** (New Values) בכרטיסיה **כללי** בחלקו התחתון של החלון **טבלה**. כדי לציין ששדה מסוג **מספור אוטומטי** צריך לקבל ערך אקראי, בחר באפשרות **אקראי** (Random) מתוך תיבת הרשימה **ערכים חדשים**.

באפשרותך להשתמש בכרטיסיות **כללי** ו**בדיקת מידע** (Lookup) כדי לבחור אפשרויות נוספות המשפיעות על השדה, כגון הנחיה למשתמש להכניס ערך בשדה, או ציון שהשדה מכיל ערך ברירת מחדל. מאפייני שדה שהוגדרו ברמת הטבלה חלים דרכה על טפסים ודוחות. מאפייני שדה טבלה יכולים גם לפשט את הקוד שנכתב עבור טפסים ודוחות. המשמעות הנוספת של תחזוקת מאפייני שדה ברמת הטבלה היא שהמאפיינים משתנים במקום אחד ולא בכל טופס ודוח שמשתמשים בשדה.

Access 2000 הוא הגירסה הראשונה של Access שמאפשרת שליטה באמצעות קוד על הערך ההתחלתי וגודל הצעד של סוגי השדה **מספור אוטומטי** (פרק 2 עוסק בהיבטים הבסיסיים של יכולת זו). באפשרותך להשתמש במילות המפתח ALTER TABLE ו-ALTER COLUMN ב- Jet SQL כדי לעדכן את ערכי ההתחלה והצעד של שדה **מספור אוטומטי**. כזכור מפרק 2, שדה מספור אוטומטי של Jet SQL כולל את סוג הנתונים IDENTITY. המאפיינים **start** ו-**step** של סוג נתונים זה מאפשרים לשנות באמצעות קוד את ערך המספור האוטומטי הבא ואת גודל הצעד של הערכים הבאים.

שדות טקסט

שדות טקסט משמשים לשמירת ערכי מחרוזת באורך של עד 255 תווים. סוג הנתונים **טקסט** (Text) יכול לאחסן פריטים כגון נתוני אנשי קשר וערכים מספריים שאינם מיועדים לחישוב (לדוגמה, מספר זהות, מספר טלפון ומספר קטלוגי). באפשרותך להשתמש בשדות **טקסט** בטבלה כדי ליצור ערכי מחרוזת מחושבים. ניתן לאינדקס מפתחות ראשיים לצורך מיון ואחזור מהירים המבוססים על שם משפחה, או על סוג שדה אחר מסוג **טקסט**.

שדות בדיקת מידע

שדה **בדיקת מידע** (Lookup) מציג ערך בעל משמעות המקביל לערך שמאוחסן בשדה (לדוגמה, מסד נתונים עשוי להשתמש במספרים ייחודיים כדי לייצג מוצר. שדה **בדיקת מידע** יכול להציג את שם המוצר במקום המספר הקטלוגי שמייצג אותו). סוג שדה זה (המוכר גם כ**עמודת מפתח**, **Key Column**) מאפשר לאחסן ערך אינדקס

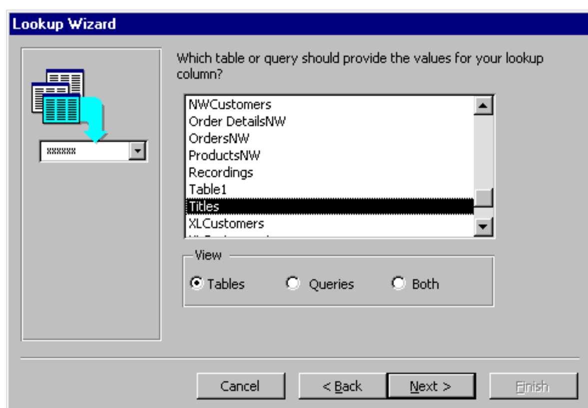
בטבלה, אך מציג ערך טקסט בעל משמעות כשהטבלה מוצגת בתצוגת **גיליון נתונים** (Datasheet). שדה **בדיקת מידע** יכול להיקשר לערכים בטבלה או בשאילתה אחרת, או לרשימת ערכים מותאמת אישית שהטבלה מתחזקת עבור שדה בדיקת המידע. לאחר הוספת שדה בדיקת מידע המפנה לערכים הנמצאים בטבלה אחרת, לא ניתן למחוק את השדה בטרם מוחקים את קשרי הגומלין לשדה האחר, באמצעות החלון **קשרי גומלין** (Relationships). בעת מחיקת קשרי הגומלין, Access יזכיר לך שהשדה מהווה עדיין חלק מקשרי הגומלין יבקש אישור לפעולת המחיקה. בשלב זה בחר את האפשרות **כן** (Yes) כדי להסיר את שדה בדיקת המידע מהטבלה.

שדה **בדיקת מידע** יוצרים באמצעות **אשף בדיקת מידע** (Lookup Wizard). בחר באשף מתיבת הרשימה הנפתחת **סוג נתונים** (Data Type) כדי לקבל טבלה בתצוגת **עיצוב** (Design). תיבת הדו-שיח ההתחלתית של האשף שואלת אם ערכי בדיקת המידע מקורם בטבלה אחרת או ברשימת ערכים מותאמת אישית. בדרך כלל תלחץ על **הבא** (Next) כדי להשתמש בערכי בדיקת המידע מטבלה אחרת.

אם ברצונך להשתמש בערכי בדיקת מידע מטבלה אחרת, באפשרותך לבחור מתוך תיבת הדו-שיח השנייה בטבלה, או בשאילתה שמכילים את הערכים המוצגים באמצעות שדה בדיקת המידע. בתיבת הדו-שיח השלישית, בחר בשדה או בשדות שמכילים את הערכים המוצגים באמצעות שדה בדיקת המידע שיצרת (בדרך כלל תבחר שדה טקסט שמשמש כמפתח ראשי).

תיבת הדו-שיח הרביעית מאפשרת לקבוע את רוחב העמודה (או העמודות) להצגת רשימת בדיקת המידע, בתצוגת **גיליון נתונים** (Datasheet). רשימת בדיקת המידע יכולה להציג את ערך שדה בדיקת המידע ופריט אחד מהרשימה לכל שדה שבחרת בטבלה הקשורה (בדרך כלל תרצה להסתיר את עמודת המפתח). תיבת הדו-שיח האחרונה מאפשרת להגדיר כיתוב עבור שדה בדיקת המידע.

תרשים 3.6 מציג את תיבות הדו-שיח השנייה, השלישית והרביעית שבאמצעותן יצרו את השדה TitleID של הטבלה Persons בקובץ מסד הנתונים של פרק 3 המופיע בתקליטור המצורף לספר.



Lookup Wizard

Which fields contain the values you want included in your lookup column? The fields you select become columns in your lookup column.

Available Fields:

Selected Fields:

TitleID
Title

Cancel | < Back | Next > | Finish

Lookup Wizard

How wide would you like the columns in your lookup column?

To adjust the width of a column, drag its right edge to the width you want, or double-click the right edge of the column heading to get the best fit.

☑ Hide key column (recommended)

Title
Mr.
Mrs.
Miss
Ms.
Dr.

Cancel | < Back | Next > | Finish

תרשים 3.6: תיבות הדו-שיח השנייה, השלישית והרביעית של **אשף בדיקת מידע** (Lookup Wizard)

הטבלה Persons בתצוגת **עיצוב** מציגה את השדה TitleID בתור שדה מספר, ולא בתור שדה **בדיקת מידע** או שדה טקסט (תרשים 3.7). סוג הנתונים של שדה **בדיקת מידע** תלוי בסוג הנתונים של עמודת המפתח. גם אם תסתיר את עמודת המפתח, סוג הנתונים שלה יקבע את סוג הנתונים של שדה בדיקת המידע.

הכרטיסייה **בדיקת מידע** בתצוגת **עיצוב** של הטבלה מכילה את משפט SQL המשמש את שדה בדיקת המידע. Access מכין אותו בעצמו בעת שאתה בוחר את האפשרויות הרצויות באמצעות אשף בדיקת המידע.

Persons : Table

Field Name	Data Type	Description
MyAutonumber	AutoNumber	AutoNumber primary key
TitleID	Number	Lookup field
FirstName	Text	
LastName	Text	
CurrencyBalance	Currency	Currency field
CurrencyFloat	Number	Number field with a Double type
CurrencyBalanceDec	Number	Number field with a Decimal type

Field Properties

General | Lookup

Display Control	Combo Box
Row Source Type	Table/Query
Row Source	SELECT Titles.TitleID, Titles.Title FROM Ti
Bound Column	1
Column Count	2
Column Heads	No
Column Widths	0";1"
List Rows	8
List Width	1"
Limit To List	Yes

The data type determines the kind of values that users can store in the field. Press F1 for help on data types.

תרשים 3.7: הטבלה Persons בתצוגת **עיצוב** ובה שדה **בדיקת מידע** שיצירתו הוצגה בתרשים 3.6

שדות מספר

שדות **מספר** (Number) שונים משדות **טקסט** (Text) מכיון שהם יכולים לקבל מיגוון של סוגי משנה, החל בבית יחיד (הסוג **בית** - Byte) וכלה ב-16 בתים (סוג המשנה **קוד העתק משוכפל** - Replication ID). סוגי משנה נוספים בטווח הסוגים הם **שלם** (Integer), **מספר שלם ארוך** (Long Integer), **יחיד** (Single), **כפול** (Double) **ועשרוני** (Decimal). למעט סוגי המשנה **בית** וקוד **העתק משוכפל**, כל השאר מתוארים בפרק 2, בסעיף "סוגי נתונים".

סוג המשנה **בית** דומה למשתנה מסוג Boolean. שני הסוגים יכולים לאחסן ערכים בוליאניים, אך הסוג **בית** דורש זיכרון בגודל בית אחד, בעוד שהסוג Boolean דורש שני בתים. סוג הנתונים **קוד העתק משוכפל** אינו זמין בתור סוג נתונים של משתנה. ייעודו העיקרי הוא לשכפול, אך הוא משמש בתור מזהה ייחודי. אורכו ושיטת יצירתו הופכים אותו לבטוח יותר לשמירה על ייחודיות לעומת שדה מספור אוטומטי.

סוג המשנה **עשרוני** מאפשר את ביטולן של שגיאות עיגול ובו בעת מאפשר לאחסן מספרים גדולים באמצעות המאפיינים **דיוק** (Precision) **וסרגל** (Scale). מאפיינים אלה שולטים במספר הספרות משני צידי הנקודה העשרונית. ערך המאפיין **דיוק** המייצג את מספר הספרות הכולל שניתן לאחסן בשדה, נע בין 1 ל-28. **סרגל**, המציין את מספר הספרות מימין לנקודה העשרונית שניתן לאחסן בשדה, נע בין 0 לערך המאפיין **דיוק**. הודות למאפיין **סרגל**, סוג המשנה **עשרוני** יכול להכיל יותר ספרות אחרי הנקודה מסוגי משנה אחרים של **מספר** מבלי לגרום לשגיאות עיגול.

תרשים 3.8 מציג את הטבלה Persons בתצוגת **גיליון נתונים**. השדה CurrencyBalance משתמש בסוג הנתונים **מטבע**, CurrencyFloat משתמש בסוג הנתונים **מספר** ובסוג המשתנה **כפול** ו-CurrencyBalanceDec משתמש בסוג הנתונים **מספר** עם סוג המשנה **עשרוני**. המאפיין **סרגל** של השדה CurrencyBalanceDec מקבל את הערך 6, שפירושו שהשדה יכול לאחסן שש ספרות מימין לנקודה העשרונית. ערך זה גבוה ממספר הספרות שיכול סוג הנתונים **מטבע** לייצג בצורה מדויקת – הוא מוגבל לארבע ספרות מימין לנקודה. סוג המשתנה **כפול** יכול לייצג מספר בעל עד שש ספרות עשרוניות, אך הוא אינו מבצע את משימתו בדיוק של מספר שלם (Integer). השורה הראשונה של הטבלה Persons מציגה את הערך 1.0001 בתבניות הנתונים **מטבע**, **כפול** ו**עשרוני**. השורה השנייה מבטאת את 1.00001 בשלוש התבניות הללו. שים לב כי התבנית **מטבע** בתצוגת **גיליון נתונים** מציגה את הערך 1.00001 בתור 1.0000, מכיון שהיא מוגבלת לארבעה מקומות עשרוניים. התבניות **כפול** ו**עשרוני** מציגות ערכים בצורה זהה.

Persons : Table

	MyIndex	TitleID	First Name	Last Name	CurrencyBalance	CurrencyFloat	CurrencyBalanceDec
▶	1	Miss	Shelly	Leghorne	$1.0001	1.0001	1.0001
	2	Mr.	Glen	Hill	$1.0000	1.00001	1.00001
*	toNumber)						

Record: 1 of 2

תרשים 3.8: הטבלה Persons בתצוגת **גיליון נתונים**, כשהיא מכילה ערכים מספריים בתבניות הנתונים **מטבע**, **כפול** ו**עשרוני**

השיגרה DecimalArithmetic שלפניך מציגה הבדלים נוספים בין תבניות הנתונים שתוארו. השיגרה מחשבת את ההפרשים בין המספרים המאוחסנים בשדות המתאימים, לבין הערך 1. השיגרה פותחת ערכת רשומות המבוססת על הטבלה Persons, מפחיתה 1 מכל אחד משלושת שדות המספר שבשורה הראשונה ומדפיסה את התוצאות בחלון **Immediate** (מיידי). לאחר מכן היא עוברת לשורה השנייה ומבצעת את התהליך פעם נוספת.

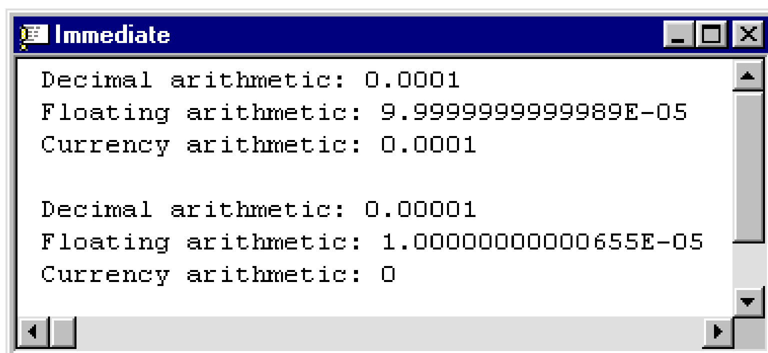
```
Sub DecimalArithmetic()

Dim cnn1 As New ADODB.Connection
Dim rst1 As Recordset
Dim intCounter As Long, sumD As Variant
Dim sumC As Variant, sumF As Variant

' Open and set recordset.
   Set rst1 = New ADODB.Recordset
   rst1.ActiveConnection = CurrentProject.Connection
   rst1.CursorType = adOpenKeyset
   rst1.LockType = adLockOptimistic
   rst1.Open "Persons", , , , adCmdTable
```

```
    Debug.Print "Decimal arithmetic: " & rst1.Fields(6) - 1
    Debug.Print "Floating arithmetic: " & rst1.Fields(5) - 1
    Debug.Print "Currency arithmetic: " & rst1.Fields(4) - 1
    rst1.MoveNext
    Debug.Print
    Debug.Print "Decimal arithmetic: " & rst1.Fields(6) - 1
    Debug.Print "Floating arithmetic: " & rst1.Fields(5) - 1
    Debug.Print "Currency arithmetic: " & rst1.Fields(4) - 1
End Sub
```

תרשים 3.9 מציג את תוצאות הפעלת השיגרה. התבניות **מטבע** ו**עשרוני** נותנות תוצאה זהה, 0.0001. התבנית **כפול** אינה מסוגלת להגיע לדיוק כזה; היא נותנת 000099999999999989.. למטרות רבות תוצאות אלו של התבניות **מטבע** ו**עשרוני** זהות. לעומת זאת, כשיש צורך בתוצאה מדויקת, אין די בכך. כשמבצעים פעולה זו על השורה השנייה שמכילה את הערך 1.00001, התבנית **עשרוני** היא היחידה שתפיק תוצאה מדויקת. חישובים אלה ממחישים את תפקידו המיוחד של סוג המשנה של סוג הנתונים **עשרוני**. עליך להשתמש בו כשדרוש דיוק ברמה שלא ניתן להשיג בתבניות הנתונים **מטבע** ו**כפול**.



תרשים 3.9: פלט השיגרה DecimalArithmetic

השדות: תזכיר, אובייקט OLE, תאריך/שעה, כן/לא

סוגי נתונים נוספים כוללים את הסוג **תזכיר** (Memo) שמאחסן מחרוזות טקסט ארוכות מאוד, אפילו יותר מ-255 תווים (המגבלה של סוג הנתונים **טקסט**). סוג הנתונים **תזכיר** יכול להגיע עד 64 ק״ב. באפשרותך לגשת אליו ולכתוב את תוכנו בקטעים של 64 ק״ב באמצעות השיטות GetChunk ו-AppendChunk, בהתאמה.

Jet 4 תומך באינדוקס של 255 התווים הראשונים של שדה **תזכיר**. הדבר שימושי במיוחד לסוגי הנתונים **היפר-קישור** (Hyperlink) התלויים בסוג הנתונים **תזכיר**.

אובייקט OLE (OLE Object) הוא סוג נתונים גדול נוסף. הוא מטפל באובייקטים בתבנית בינארית, כגון חוברת עבודה של Microsoft Excel או מסמך של Microsoft Word.

סוגי הנתונים **תאריך/שעה** (Date/Time) יכולים לייצג גם תאריכים וגם שעות. ערכי תאריך מאוחסנים משמאל לנקודה העשרונית; ערכי שעה מאוחסנים לימינה (עיין בדוגמאות הקוד של פרק 2 המטפלות בערכי שדה **תאריך/שעה**).

סוג הנתונים **כן/לא** (Yes/No) הוא הקטן ביותר. הוא מופיע באחד משני המצבים: כן/לא, אמת/שקר (True/False) או **מופעל/מכובה** (On/Off). סוג זה תופס זיכרון בגודל בית אחד בלבד.

אימות נתונים

לעיצוב טבלאות עבור מסד נתונים חסין לתקלות, עליך להבטיח כי רק נתונים חוקיים יוכלו להיכנס למסד הנתונים. Access 2000 כולל כמה רכיבים שיסייעו לך להשיג זאת.

המאפיינים נדרש ואפשר אורך אפס

לעיתים רשומה אינה חוקית, אלא אם היא כוללת ערך שדה מסוים, כגון שדה מפתח ראשי או שדה מפתח זר. קביעת המאפיין **נדרש** (Required) של מפתח זר בתור **כן** (Yes) מבטיח כי המשתמשים לא יוכלו להיכנס לרשומה בצד ה"רבים" של קשר הגומלין מבלי שתהיה התאמה לרשומה אחת לפחות בצד הנגדי של קשר הגומלין.

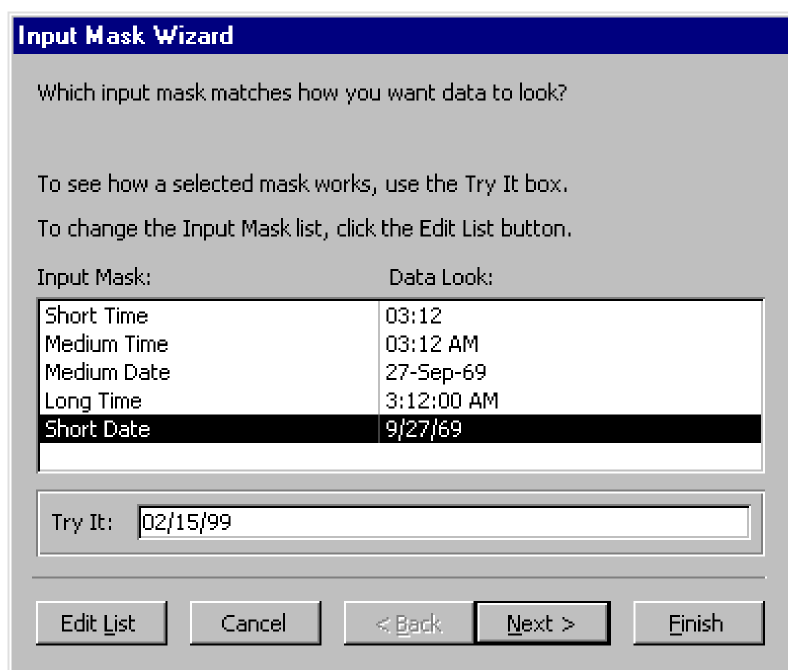
כשנותנים למאפיין **נדרש** של שדה את הערך **כן**, Access אינו מקבל רשומה שערך השדה שלה Null. הגדרת המאפיין **אפשר אורך אפס** (Allow Zero Length) מאפשרת (או לא) להציב בשדה מחרוזת באורך אפס (""). Access מבדיל בין שדה שלא קיבל ערך כלשהו (Null) לבין שדה שאינו מכיל ערך (מחרוזות באורך אפס בשדות **טקסט**).

המאפיין מסכת קלט

מסכת קלט (Input Mask) היא תבנית שמנחה את המשתמש בנושא סוג ותבנית הנתונים הדרושים. בדומה למאפייני שדה אחרים, מאפיין זה חל על טפסים ודוחות באמצעות השימוש בשדה.

באפשרותך להשתמש במסכות קלט תקניות, או ליצור בעצמך מסכות קלט מותאמות אישית. תרשים 3.10 מציג את **אשף מסכות הקלט** המציג מבחר מסכות קלט של השדה **תאריך/שעה** (Date/Time). באפשרותך להכניס ערכים בתיבת הטקסט **ניסיון** (Try It) כדי לראות כיצד תתפקד המסכה במצבים של נתוני אמת.

האשף קובע את קלט השדה ואת תצוגת נתוני השדה, אלא אם תציין גם את המאפיין **תבנית-עיצוב** (Format) של שדה. כשיישום מגדיר את המאפיין **תבנית-עיצוב** של שדה, המאפיין קובע את תצוגת נתוני השדה. המאפיין **תבנית-עיצוב** אינו משפיע על התצוגה של ערך, אלא רק לאחר שהיישום שומר את הערך במסד הנתונים.



תרשים 3.10: אשף מסכות הקלט מציג מבחר מסכות אפשריות עבור שדות מהסוג **תאריך/שעה**

המאפיינים חוק אימות וטקסט אימות

מאפיינים אלה נועדו להבטיח כי המשתמש מקליד מידע חוקי. המאפיין **חוק אימות** (Validation Rule) נועד להגדרת קריטריוני חוקיות לערכים בשדה (אם שדה אמור לקלוט נתונים גדולים מ-1, למשל, עליך להגדיר את חוק אימות בתור 1<). באפשרותך להשתמש במאפיין **טקסט אימות** (Validation Text) כדי להגדיר הודעות תגובה שתוצגנה למשתמשים בעת שינסו להקליד ערך מחוץ לטווח הערכים החוקיים שנקבע במאפיין חוק אימות.

באפשרותך לציין גם כללי אימות עבור הטבלה כיחידה אחת. לשם כך, פתח את הטבלה בתצוגת **עיצוב** ולחץ על לחצן **מאפיינים** (Properties) בסרגל הכלים **עיצוב טבלה** (Table Design). לאחר מכן השתמש במאפיינים **חוק אימות** ו**טקסט אימות** כדי להגדיר כלל אימות עבור הטבלה בשלמותה והודעה שתוצג בתגובה להפרת הכלל. כלל האימות של טבלה מאפשר להגדיר קריטריונים שהיקף ההחלה שלהם משתרע מעבר לשדה יחיד.

אם דרושים לך מערכי כללים רבים, תוכל לכלול אותם בביטוי כלל האימות באמצעות פסוקיות And (וגם) (אם שדה אמור לקלוט רק מספרים גדולים מ-1 וקטנים מ-10, באפשרותך להגדיר את **חוק אימות** בתור 10> And 1<). לחץ על לחצן **בנייה** (Build) הסמוך לתיבת המאפיין **חוק אימות** כדי לפתוח את **בונה ביטויים**

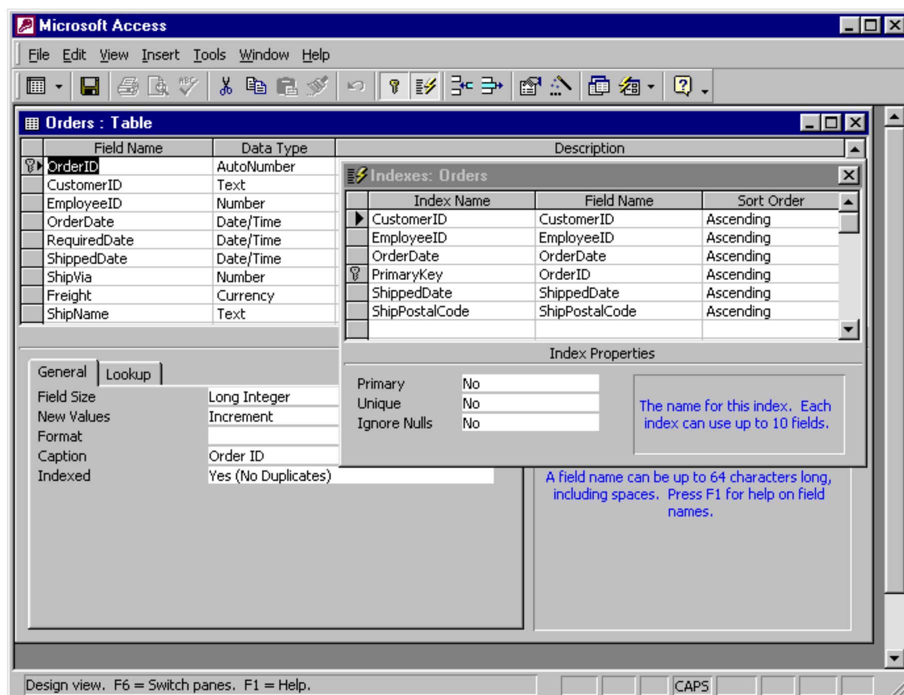
(Expression Builder). באפשרותך להשתמש בפונקציות המוכללות של Access לאימות שדות טבלה, אך פונקציות מותאמות אישית אסורות לשימוש בביטויי אימות של טבלה או של שדה. **בונה ביטויים** מתאים ליצירת אימות טבלה כולל בדיוק כפי שהוא מתאים לביטויי אימות של שדה.

יצירת אינדקסים

אינדקסים קובעים את אופן התפקוד של טבלאות ואת היחסים ביניהן. אינדקסים מאיצים פעולות מיון, חיפוש ובחירה ברמת השדה. למרות שהשימוש באינדקסים פוגע בביצועי יישומים בעת קליטת נתונים (מכיון שאינדקסים דורשים הצבת ערכים בכל רשומה חדשה), יתרונותיו עולים על חסרונותיו.

אינדקסים תומכים גם בשלמות הקשרים. לפחות אחד מהשדות המקשרים בקשר גומלין חייב להיות מפתח ראשי או אינדקס ייחודי.

תרשים 3.11 מציג את הטבלה Orders בתצוגת **עיצוב**, כשהחלון הראשי וחלון האינדקסים פתוחים. הלחצן **אינדקסים** (Indexes) שבסרגל הכלים **עיצוב טבלה** (Table Design) מאפשר להציג או להסתיר את חלון האינדקסים.



תרשים 3.11: החלון הראשי והחלון **אינדקסים** של הטבלה Orders בתצוגת **עיצוב**

החלון **אינדקסים** שבתרשים 3.11 כולל גם שורה ובה מפתח. בדיוק כמו בחלון הראשי, שורה זו מסמנת את המפתח הראשי. בחלון האינדקסים נבחר האינדקס CustomerID.

אינדקס זה תלוי בשדה CustomerID שהוא מפתח זר בטבלה Orders. האינדקס CustomerID מקשר בין הטבלאות Customers ו-Orders בקשר גומלין מסוג יחיד לרבים. הוא אינו המפתח הראשי של הטבלה Orders, ולכן ניתן לשכפל אותו בין רשומות. שים לב גם כי כמה אינדקסים, כגון OrderDate ו-ShipPostalCode, אינם מפתחות של טבלאות אחרות. אחת המטרות ליצירת אינדקס היא האצת הפעולות המתבצעות בנתוני הטבלה, כגון בחירה לפי תאריך, או מיון בהתאם לקוד דיוור.



הערה:
אחת הדרכים להבטיח את תקינות ההגדרות של מאפיין מפתח זר היא להעתיק את השדה אל **הלוח** (Clipboard) של Windows בצד ה"יחיד" של קשר הגומלין. פעולה זו מאפשרת להדביק את השדה בתצוגת **עיצוב** של הטבלה בצד ה"רבים" של קשר הגומלין. כך ניתן להימנע מהצורך להגדיר ידנית את מאפייני המפתח הזר.

אינדקס יוצרים באמצעות הקלדת שם בעמודה **שם אינדקס** (Index Name) בחלון **אינדקסים**. לאחר מכן בוחרים שדה עבור האינדקס מתוך תיבת הרשימה הנפתחת, ובוחרים את סדר המיון של השדה: עולה או יורד. אם לאינדקס שייך שדה נוסף, בוחרים את שמו ואת סדר המיון בשורה הבאה מייד לאחר מכן. באפשרותך להוסיף שדות נוספים לאינדקס באופן זהה. כל שדה שתוסיף לאינדקס חייב להופיע מייד מתחת לשדה הקודם. רק השדה הראשון של האינדקס צריך להכיל ערך עבור העמודה **שם אינדקס**. כל השורות הבאות לאחר מכן שעמודת שם האינדקס שלהן ריקה, שייכות לאותו אינדקס. כל ערך בעמודה **שם אינדקס** פותח אינדקס חדש.

באפשרותך להגדיר את שלושת מאפייני האינדקס בנפרד. יחד עם זאת, אם תגדיר את **ראשי** (Primary) בתור **כן** (Yes), המאפיינים **ייחודי** (Unique) ו**התעלם מ-Nulls** (Ignore Nulls) יקבלו את הערכים **כן** ו**לא**, בהתאמה. מפתח ראשי חייב להיות ייחודי לכל רשומה. לא ניתן ליצור מפתח ראשי בשדה או בשדות שמכילים ערכי Null. לטבלה נתונה יכול להיות מפתח ראשי אחד בלבד. טבלה יכולה להכיל שדות רבים בעלי אינדקסים ייחודיים שמתעלמים מערכי Null. אינדקס כלשהו, למעט מפתח ראשי, יכול להתעלם מערכי Null. הדבר מאפשר להגדיר אינדקס שמבוסס על ערכי שדה שונים מ-Null. הבחירה להתעלם מערכי Null יכולה לצמצם את דרישות האחסון של אינדקס.

תרשים 3.12 מציג את החלון הראשי וחלון האינדקסים של הטבלה Order Details בתצוגת **עיצוב**. שים לב שבחלון הראשי נבחר השדה ProductID. הכיתוב שלו הוא Product, והוא אחד משני השדות המגדירים את המפתח הראשי. השדה השני הוא OrderID. שני השדות הם שדות **מספר** (Number) שסוג הנתונים שלהם הוא **מספר שלם ארוך** (Long Integer); אף אחד מהם אינו שדה **מספור אוטומטי** (AutoNumber), אך לעומת זאת שניהם משמשים בתור מפתחות זרים המבוססים על שדות מספור אוטומטי בטבלאות Orders ו-Products. שניהם יחד מזהים באופן ייחודי כל שורה בטבלה Order Details. זוהי התנהגות אופיינית לטבלאות תיבת הסתעפויות.

חלון האינדקסים בתרשים 3.12 מוסיף ומזהה את המפתח הראשי. שים לב כי הטבלה Order Details מקיימת גם אינדקסים נפרדים שמבוססים על השדות OrderID ו-ProductID. אינדקסים אלה מקיימים קישורים עם הטבלאות Orders ו-Products. שלמות הקשר בין Order Details ו-Products מבוססת על אינדקסים אלה.

Order Details : Table

	Field Name	Data Type
🔑	OrderID	Number
🔑	ProductID	Number
	UnitPrice	Currency
	Quantity	Number
	Discount	Number

Indexes: Order Details

	Index Name	Field Name	Sort Order
	OrderID	OrderID	Ascending
🔑	PrimaryKey	OrderID	Ascending
🔑		ProductID	Ascending
	ProductID	ProductID	Ascending

Index Properties

Primary	Yes
Unique	Yes
Ignore Nulls	No

The name for this index. Each index can use up to 10 fields.

General | Lookup

Field Size	Long Integer
Format	
Decimal Places	Auto
Input Mask	
Caption	Product
Default Value	
Validation Rule	
Validation Text	
Required	Yes
Indexed	Yes (Duplicates OK)

The data type determines the kind of values that users can store in the field. Press F1 for help on data types.

תרשים 3.12: החלון הראשי והחלון **אינדקסים** של הטבלה Order Details של מסד הנתונים Northwind

באפשרותך ליצור ולנהל ידנית יחסי שלמות קשרים בין טבלאות מתוך החלון **קשרי גומלין** (Relationships).

1. הוסף את הטבלאות לחלון, במידה שלא עשית זאת עד כה. לחץ לחיצה ימנית בחלון, בחר באפשרות **הצגת טבלה** (Show Table) והוסף לחלון טבלאות לפי הצורך.
2. צרף את הטבלאות באמצעות קישור השדות המשותפים של הטבלאות. עשה זאת על ידי גרירת שדה אחד או יותר מטבלה לטבלה, מצד ה"ייחיד" לצד ה"רבים" של קשר הגומלין.
3. בחר את שורת הצירוף.
4. לחץ לחיצה ימנית על הקו המקשר ובחר את **עריכת קשר גומלין** (Edit Relationships).

5. סמן את תיבת הסימון **אכוף שלמות הקשרים בין הטבלאות** (Enforce Referential Integrity). בצע את שינויי העיצוב הדרושים בקשר הגומלין, כגון בחירת תיבות סימון לעדכונים ומחיקות בהתאם להיררכית הקשרים. באפשרותך גם ללחוץ על **סוג צירוף** (Join Types) כדי לבחור אחד משני סוגי הצירוף נוסף על הצירוף הרגיל הכולל שורות משתי הטבלאות רק כששדות הצירוף בשתי הטבלאות זהים. שתי האפשרויות האחרות כוללות את כל הרשומות מצד ה"יחיד", או כל הרשומות מצד ה"רבים" של קשר הגומלין.

תרשים 3.13: שתי תיבות הדו-שיח **עריכת קשר גומלין** המתארות את קשרי הגומלין של שלמות קשרים בין הטבלאות Products ו- Order Details (העליונה) ובין הטבלאות Orders ו- Orders Details (התחתונה) של מסד הנתונים Northwind

תרשים 3.13 מציג את תיבות הדו-שיח **עריכת קשר גומלין** שמגדירות שלמות קשרים בין הטבלה Order Details לבין הטבלה Orders והטבלה Products. שים לב ש-Access מפרש את שתי מערכות קשרי הגומלין בתור יחיד לרבים. קשר הגומלין בין הטבלאות Orders ו- Order Details מגדיר מחיקות בהתאם להיררכית הקשרים. הדבר מאפשר למחוק הזמנה ואת כל שורות הפריטים שלה בפעולה אחת. ללא מנגנון זה היה על היישום לבצע שתי שאילתות מחיקה, אחת לכל טבלה. קשר הגומלין בין הטבלאות Products ו- Order Details אינו כולל מחיקות בהתאם להיררכית הקשרים. במקרה זה, לא תתבצע מחיקה אוטומטית של שורות פריט בהזמנות כשיתעורר צורך להסיר מוצר מהטבלה Products. המודל העסקי עשוי לחייב מאמץ מיוחד לרכישת המוצר עבור ההזמנות הפתוחות שטרם סופקו.

יצירה וניהול של טבלאות באמצעות קוד

לפעמים – ואולי אפילו ברוב הפעמים – תעדיף ליצור טבלאות באופן ידני. לעומת זאת, לעיתים תמצא כי נוח יותר ליצור טבלה באופן דינמי, כמו במקרה שעליך לשמור תוצאות ביניים לשימוש חוזר בפעולה הבאה. Access 2000 מאפשר לעשות זאת באמצעות טכניקות של ADO וגם בטכניקות מקובלות יותר של Data Access) DAO Objects). בסעיף זה נתאר יצירה וניהול של טבלאות באמצעות טכניקות ADO. גישה זו תאפשר לך לנצל את הייתרונות של חידושי הטכנולוגיה העתידית של Microsoft לגישה לנתונים.

לאחר שיוצרים טבלה באמצעות קוד, לעיתים קרובות יש צורך למלא אותה בנתונים. אף כאשר יוצרים טבלה בדרך אחרת, מילויה בנתונים באמצעות קוד הוא רעיון מצודד. ניתן לנצל מקורות נתונים רבים כדי למלא טבלה של Access בנתונים. בסעיף זה נסקור בהרחבה את השימוש בטבלה נוספת של Access, טבלה של Excel (בתור דוגמה למקור נתונים מסוג ISAM), ומקור נתונים ODBC. נלמד גם כיצד למלא טבלה של Access בנתונים באמצעות ספקי מסד נתונים OLE (OLE DB) והאובייקט הפנימי DoCmd של Access, המאפשר ליצור קישור אל מקורות נתונים מרוחקים בצורה קלה ופשוטה. בניגוד לשימוש ב- OLE DB, DoCmd הופך את מקור הנתונים המקושר לזמין מתוך החלון **מסד נתונים** (Database).

יצירת טבלה

כדי להוסיף טבלה באופן דינמי, כל שדרוש לך הוא ADO – הספריה ADOX, ליתר דיוק. בדיוק כפי שיש לתת שם לטבלה, בין אם יוצרים אותה באופן ידני או דינמי, יש גם להוסיף לה עמודות ולצרף אותה לאוסף Tables.

השיגרה MakeLocalTable שלפניך, יוצרת טבלה באופן דינמי. השיגרה פותחת בהכרזה על האובייקט Catalog והאובייקט Table. האובייקט Catalog הוא מכולה של סכמת מסד הנתונים, כולל האוסף Tables. לאחר מכן, השיגרה יוצרת מופע של האובייקטים Catalog ו-Table. השיגרה נותנת לטבלה החדשה את השם FamilyMembers2 ומוסיפה

לה ארבע עמודות באמצעות ארבע שיטות Append מקוננות בתוך משפט With...End. כל אחת מהשיטות Append כוללת שם עמודה, קבוע שמגדיר סוג נתונים וארגומנט אורך, במידת הצורך. כל אחד משלושה שדות הטקסט (אלה שנוצרו על ידי הגדרת סוג הנתונים adVarWChar) כולל פרטים על גודל השדה. השיגרה מסתיימת בצירוף הטבלה המושלמת לאוסף Tables של הקטלוג ובסגירת האובייקט Catalog.

```
Sub MakeLocalTable()
Dim cat1 As ADOX.Catalog
Dim tbl1 As ADOX.Table

'Reference objects for table
    Set cat1 = New Catalog
    cat1.ActiveConnection = CurrentProject.Connection
    Set tbl1 = New Table

' Name table and append columns.
    With tbl1
        .Name = "FamilyMembers2"
        .Columns.Append "FamID", adInteger
        .Columns.Append "Fname", adVarWChar, 20
        .Columns.Append "Lname", adVarWChar, 25
        .Columns.Append "Relation", adVarWChar, 30
    End With

' Append new table to Tables collection
'and free catalog resource.
    cat1.Tables.Append tbl1
    Set cat1 = Nothing
End Sub
```

הטבלה שלפניך מציגה את סוג הנתונים ואת סוגי המשנה המיוצגים באמצעות קבועי המחלקה DataTypeEnum של הספריה ADOX.

קבועי סוג עמודה וסוגי הנתונים הידניים השקולים להם

קבוע	ערך	סוג נתונים בקלט ידני
adBoolean	11	כן/לא (Yes/No)
asCurrency	6	מטבע (Currency)
adDate	7	תאריך/שעה (Date/Time)
adDecimal	14	מספר – עשרוני (Decimal)
adDouble	5	מספר – כפול (Double)
adGuid	72	מספר – קוד העתק משוכפל (Replication ID)

קבוע	ערך	סוג נתונים בקלט ידני
adInteger	3	מספור אוטומטי (AutoNumber)
adInteger	3	מספר – מספר שלם ארוך (Long Integer)
adLongVarBinary	205	אובייקט OLE (OLE Object)
adLongVarWChar	203	היפר-קישור (Hyperlink)
adLongVarWChar	203	תזכיר (Memo)
adSingle	4	מספר – יחיד (Single)
adSmallInt	2	מספר – שלם (Integer)
adUnSignedTinyInt	17	מספר – בית (Byte)
adWChar	130	טקסט (Text)

הטבלה מציגה כמה נקודות מעניינות. ראשית, סוג הנתונים **היפר-קישור** שקול לסוג הנתונים **תזכיר** מנקודת מבט של תכנות. שנית, לא קיים סוג נתונים נפרד לשדות מספור אוטומטי בספריה ADOX. אם הקוד בודק את המאפיין **Type** (סוג נתונים) של שדה מספור אוטומטי, תקבל את הערך adInteger. ערך זה אינו משקף את טבעו הדינמי של סוג הנתונים מספור אוטומטי. Jet 4, לעומת זאת, כולל סוג נתונים נפרד שנקרא Identity, שמקביל לסוג הנתונים מספור אוטומטי (לקבלת מידע נוסף אודות השימוש בסוג נתונים זה, עיין בפרק 2). הסעיף "יצירת טבלה באופן ידני" שהופיע קודם לכן בפרק, מתאר מילות מפתח נוספות לניהול שדות מספור אוטומטי באמצעות קוד.

הימנעות מהחלפת טבלה

בעת הוספה ידנית של טבלה למסד נתונים, קל לבדוק אם שם הטבלה קיים כבר. אם תנסה בשוגג לשמור במסד הנתונים טבלה בשם של טבלה שכבר קיימת, Access יזהיר אותך מכך וישאל אם ברצונך לדרוס את הטבלה הקיימת. לעומת זאת, אם שיגרה מנסה ליצור באמצעות קוד, טבלה בשם של טבלה קיימת, Access VBA נעצר ומציג שגיאת זמן ריצה. עליך ליצור מנגנון לכידת שגיאות במצב כזה. קיימות כמה גישות לטיפול במצב זה. השיטה ההולמת ביותר תלויה בתדירות שבה תיצור טבלאות חדשות.

השיגרה MakeLocalTableErrCatcher שלפניך מבוססת על גישה קלאסית ללכידת שגיאות. ראשית, היא מאפשרת את השימוש בשיגרה מותאמת אישית לטיפול בשגיאות, כך שהתוכנית יכולה לנהל שגיאות. המשפט On Error בתחילת השיגרה אחראי על כך. לאחר מכן, השיגרה מנסה ליצור ולצרף את הטבלה FamilyMembers2 (בדומה לשיגרה MakeLocalTable). אם הטבלה קיימת כבר בקטלוג, Access VBA יוצר שגיאה (-2147217857) ומעביר את השליטה לרוטינת הטיפול בשגיאות TableErrCatcher. לוגיקת לכידת השגיאות מחפשת את השגיאה "כבר קיים". אם היא

מזהה אותה, היא מוחקת את הטבלה הקיימת ומחזירה את השליטה לשורה שגרמה לשגיאה. הדבר מאפשר לתוכנית לשמור את הטבלה החדשה ולצאת מהשיגרה בצורה מסודרת. אם שגיאה אחרת גרמה לתוכנית לעבור דרך רוטינת הטיפול בשגיאות, השיגרה מדפיסה את מספר השגיאה בחלון **Immediate** (מיידי) בטרם תסתיים בצורה מסודרת. לא יקרה מצב שהשיגרה תסתיים בהצגת הודעת מערכת.

```
Sub MakeLocalTableErrCatcher()
On Error GoTo TableErrCatcher
Dim cat1 As ADOX.Catalog
Dim tbl1 As ADOX.Table

'Reference objects for table
    Set cat1 = New Catalog
    cat1.ActiveConnection = CurrentProject.Connection
    Set tbl1 = New Table

' Name table and append columns.
    With tbl1
        .Name = "FamilyMembers2"
        .Columns.Append "FamID", adInteger
        .Columns.Append "Fname", adVarWChar, 20
        .Columns.Append "Lname", adVarWChar, 25
        .Columns.Append "Relation", adVarWChar, 30
    End With

' Append new table to Tables collection and free catalog resource.
    cat1.Tables.Append tbl1
    Set cat1 = Nothing

' Exit the procedure.
TableErrExit:
    Exit Sub

    TableErrCatcher:
' Trap "table already exits" error.
' Delete table and resume.
    If Err.Number = -2147217857 Then
        cat1.Tables.Delete "FamilyMembers2"
        Resume
    End If

' Print details for other errors and exit.
    Debug.Print Err.Number, Err.Description

    Resume TableErrExit
End Sub
```

החלפת טבלה

אם יישום מסד הנתונים יוצר באופן קבוע את הטבלה FamilyMembers2, השיגרה עלולה לגרום לשגיאה מהסוג "כבר קיים" בכל פעם שמפעילים אותה. במצב זה, השיגרה תפעל מהר יותר אם תנסה למחוק את הטבלה הקיימת בטרם תצרף את הטבלה החדשה. כך ניתן להימנע מהצורך לטפל בשגיאה. עדיין יש צורך בשגרת טיפול בשגיאות למצבים שטבלה אינה קיימת, או למצבי שגיאה אחרים. השיגרה שלפניך כותבת את מנגנון הטיפול בשגיאות כשמוחקים ללא תנאי טבלה בעלת שם זהה לזו שמנסים לצרף למסד הנתונים.

```
Sub MakeLocalTableErrCatcher2()
On Error GoTo TableErrCatcher
Dim cat1 As ADOX.Catalog
Dim tbl1 As ADOX.Table
' Reference objects for table
   Set cat1 = New Catalog
   cat1.ActiveConnection = CurrentProject.Connection
   Set tbl1 = New Table
' Name table and append columns.
   With tbl1
      .Name = "FamilyMembers2"
      .Columns.Append "FamID", adInteger
      .Columns.Append "Fname", adVarWChar, 20
      .Columns.Append "Lname", adVarWChar, 25
      .Columns.Append "Relation", adVarWChar, 30
   End With
' Delete the old table (if it is there).
' Append the new one, and free the resource.
   cat1.Tables.Delete "FamilyMembers2"
   cat1.Tables.Append tbl1
   Set cat1 = Nothing
' Exit the procedure.
TableErrExit:
   Exit Sub
TableErrCatcher:
'Trap "object not in collection" error.
' Resume at next line.
   If Err.Number = 3265 Then
      Resume Next
   End If
' Print details for other errors and exit.
   Debug.Print Err.Number, Err.Description
   Resume TableErrExit
End Sub
```

שגיאה 3265 הנזכרת בשגרת הטיפול בשגיאות של MakeLocalTableErrCatcher2, היא תוצאה של הניסיון למחוק אובייקט שלא נמצא באוסף. השיגרה MakeLocalTableErrCatcher2 פשוט לוכדת את השגיאה וממשיכה להתבצע לאחר השורה שגרמה לה. כל שגיאה אחרת גורמת לסיום מסודר של התוכנית, כשהסימן היחיד לשגיאה הוא מספר השגיאה ותיאורה בחלון **Immediate**.

עבודה עם אינדקסים

באפשרותך להוסיף גם מפתחות ראשיים, אינדקסים וקשרי גומלין באמצעות קוד. תוכל להגדיר מפתחות ראשיים ואינדקסים בשדה יחיד או בשדות רבים.

יצירת מפתח ראשי

הוספת מפתח ראשי או אינדקס לטבלה כמוה כהוספת טבלה חדשה לקטלוג. ראשית, עליך ליצור הֶקְשֶׁר שאליו יתוסף האינדקס, כולל קטלוג וטבלה. שנית, עליך להגדיר את מאפייני האינדקס. אלה עשויים להשתנות בין אינדקסים ומפתחות ראשיים. שלישית, עליך להוסיף עמודה לאינדקס ולאחר מכן לצרף את האינדקס החדש לטבלה. אם חלה שגיאה, כגון אם האינדקס כבר קיים, עליך לתת מענה הולם לכך. השיגרה AddPK שלפניך יוצרת מפתח ראשי באופן דינמי.

```
Sub AddPK()
Dim cat1 As New ADOX.Catalog
Dim tbl1 As New ADOX.Table
Dim pk1 As New ADOX.Index
' Create a context for the new primary key.
   cat1.ActiveConnection = CurrentProject.Connection
   Set tbl1 = cat1.Tables("FamilyMembers2")
' Set the primary key properties.
   With pk1
       .Name = "MyPrimaryKey"
       .PrimaryKey = True
       .Unique = True
       .IndexNulls = adIndexNullsDisallow
   End With

' Append column to index and index to table.
   pk1.Columns.Append "FamID"
   tbl1.Indexes.Append pk1

' Free resources.
   Set cat1 = Nothing
End Sub
```

השיגרה פותחת בהכרזה ויצירת מופע של האובייקטים Catalog, Table ו-Index (ליצירת מפתח ראשי יש צורך בכל שלושת האובייקטים). שים לב שלא קיים אובייקט מפורש עבור מפתח ראשי. בשלב הבא, השיגרה מגדירה את ההקשר להגדרת מפתח ראשי חדש. השיגרה מגדירה את המאפיין ActiveConnection של האובייקט Catalog כך שהקטלוג יצביע על מסד נתונים מסוים. לאחר מכן היא מגדירה את הפניית הטבלה לטבלה שנמצאת במסד נתונים זה. הפניה זו היא הטבלה שאליה תוסיף השיגרה שכתבת את המפתח הראשי החדש.

כעת, השיגרה מגדירה ארבעה מאפייני אינדקס. הראשון הוא שם המפתח הראשי. הוא מופיע בתור ערך בעמודה **שם אינדקס** (Index Name) בחלון **אינדקסים** (Indexes) של הטבלה. שלושת המאפיינים הנותרים מייחדים את המפתח הראשי מאינדקס פשוט כלשהו. עליך להקפיד ולהגדיר מאפיינים אלה כפי שהם מופיעים בשיגרה AddPK בעת יצירת מפתח ראשי.

בשלב זה, מפעילה השיגרה שתי שיטות Append. הראשונה מצרפת את העמודה FamID מתוך הטבלה FamilyMembers2 אל האינדקס. השיטה השנייה מצרפת את האינדקס לטבלה. לבסוף, השיגרה מגדירה את האובייקט Catalog בתור Nothing, ועל ידי כך משחררת את המשאבים ששימשו ליצירת המפתח הראשי.

לעיתים קרובות יש צורך בשדה שסוג הנתונים שלו הוא מספור אוטומטי, מספר שלם ארוך או טקסט, שישמש כמפתח ראשי. דבר זה משפר את ביצועי מסד הנתונים בהשוואה למפתח ראשי המבוסס על שדות רבים. לעומת זאת, יש מקרים שהשימוש באינדקס המבוסס על שדות רבים הגיוני יותר במונחים של אופן השימוש בנתוני הטבלה. במקרה שאינדקס מבוסס-שדות רבים הולם את הצרכים, כל שעליך לעשות הוא לצרף יותר מעמודה אחת לאינדקס בטרם תצרף את האינדקס לטבלה.

בדוגמה לעיל, ניתן להחליף את השורה:

```
pk1.Columns.Append "FamID"
```

בשורות הבאות:

```
pk1.Columns.Append "Lname"
pk1.Columns.Append "Fname"
pk1.Columns.Append "Relation"
```

שורות אלו מגדירות מפתח ראשי בשלושה שדות במקום בשדה אחד. השיגרה AddPK יוצרת מפתח ראשי המבוסס על שדה יחיד; החלון **אינדקסים** הראשון בתרשים 3.14 מציג את תוצאת פעולת השיגרה AddPK. החלון השני בתרשים מציג את תוצאת פעולת השיגרה AddPK3 – שיגרה זהה ל- AddPK, למעט השינוי בשורות הקוד (שתי השגרות מופיעות בתקליטור המצורף לספר, בחלק של פרק 3). לא ניתן לקיים אלא מפתח ראשי יחיד בזמן נתון, ולכן עליך להסיר ידנית את המפתח הראשי בין הקריאה לשגרות AddPK ו-AddPK3.

Indexes: FamilyMembers2

Index Name	Field Name	Sort Order
MyPrimaryKey	FamID	Ascending

Index Properties

Primary	Yes
Unique	Yes
Ignore Nulls	No

The name for this index. Each index can use up to 10 fields.

Indexes: FamilyMembers2

Index Name	Field Name	Sort Order
MyPrimaryKey	Lname	Ascending
	Fname	Ascending
	Relation	Ascending

Index Properties

Primary	Yes
Unique	Yes
Ignore Nulls	No

The name for this index. Each index can use up to 10 fields.

תרשים 3.14: חלונות **אינדקסים** לאחר הפעלת השגרות AddPK ו-AddPK3

השגרות AddPK ו-AddPK3 יכולות להיכשל מסיבות אחדות. השיגרה AddPKErr שלפניך לוכדת שתי שגיאות: מפתח ראשי שכבר קיים (קוד שגיאה 2147467259-) וטבלה שכבר נפתחה (קוד שגיאה 2147217856-). כפי שנאמר, לא ניתן להוסיף מפתח ראשי חדש אם קיים כבר מפתח ראשי. בנוסף, לא ניתן לשנות את מבנה האינדקס אם הטבלה פתוחה.

```
Sub AddPKErr()
On Error GoTo PKErr
Dim cat1 As New ADOX.Catalog
Dim tbl1 As New ADOX.Table
Dim pk1 As New ADOX.Index
Dim iNumber As Integer
' Create a context for the new primary key.
   cat1.ActiveConnection = CurrentProject.Connection
   Set tbl1 = cat1.Tables("FamilyMembers2")
```

```
' Set the primary key properties.
' The label (SetPKvariable) gives the procedure a recovery point
' from a previously existing primary key.
SetPKvariable:
    With pk1
        .Name = "MyPrimaryKey"
        .PrimaryKey = True
        .Unique = True
        .IndexNulls = adIndexNullsDisallow
    End With
' Append column to index and index to table.
    pk1.Columns.Append "FamID"
    tbl1.Indexes.Append pk1
' Exit procedure.
PKErrExit:
    Set cat1 = Nothing
    Exit Sub
PKErr:
' Checks for table already in use
    If Err.Number = -2147217856 Then
        MsgBox "FamilyMembers2 currently in use.  This" & _
            " operation requires the table to be closed."
' Checks for primary key already exists
    ElseIf Err.Number = -2147467259 Then
        For Each pk1 In tbl1.Indexes
            If pk1.PrimaryKey = True Then
                tbl1.Indexes.Delete (iNumber)
                Resume SetPKvariable
            End If
            iNumber = iNumber + 1
        Next pk1
' Traps for other errors
    Else
        MsgBox "Open Immediate window for Bug report"
        Debug.Print Err.Number; Err.Description
        Resume PKErrExit
    End If
End Sub
```

חוץ מכמה חריגות, הלוגיקה של השיגרה AddPKErr זהה ברובה לזו של AddPK. AddPKErr מאפשרת להפעיל תחילה רוטינה לטיפול בשגיאות באמצעות המשפט On Error. אזור ההכרזה כולל משפט Dim חדש למשתנה מסוג Integer. מנגנון לכידת השגיאות מנצל משתנה זה בתור אינדקס בלולאה שעוברת על האוסף Indexes של הטבלה.

מנגנון לכידת השגיאות של AddPKErr מתחיל את פעולתו בתווית PKErr. משפט If...Then בודק בשלב ראשון אם הטבלה פתוחה. אם הטבלה פתוחה, השיגרה מציגה תיבת הודעה שמסבירה את הבעיה ומציעה פתרון לפני שתסתיים בצורה מסודרת. הפסוקית ElseIf מזהה את קיומו של מפתח ראשי. במקרה זה, הרוטינה ממספרת את האינדקסים בטבלה, עד שמגיעה לאינדקס שהמאפיין PrimaryKey שלו מוגדר בתור True. כעת היא מוחקת את האינדקס ומחזירה את השליטה לשלב ההתחלתי של הגדרת מפתח ראשי חדש. פעולה זו הכרחית, מכיון שהשגיאה מבטלת את ההגדרות הקודמות. אם הטבלה אינה פתוחה וגם לא קיים מפתח ראשי כלשהו, השיגרה כותבת את מספר השגיאה ואת תיאורה לחלון **Immediate** לפני שמסתיימת. מוצגת למשתמש תיבת הודעה שמורה לו להציג את החלון **Immediate** לקבלת פרטי סיבת השגיאה.

יצירת אינדקס

הוספת אינדקס פשוט לטבלה אינה שונה בהרבה מהוספת מפתח ראשי. ההבדל העיקרי הוא שאין מגדירים את המאפיין PrimaryKey בתור True (כברירת מחדל, ערכו הוא False). השיגרה AddIdx שלפניך דומה במבנה שלה ל-AddPK. מלבד זה שאינה מגדירה את מאפיין המפתח הראשי, ההבדל העיקרי ביניהן הוא שהיא קובעת מפורשות סדר מיון לאינדקס. השיגרה ממיינת את הטבלה לפי העמודה FamID בסדר יורד, ועל ידי כך גורמת לה להציג קודם את הרשומות המאוחרות (בהנחה שרשומת בעלות ערך FamID גדול יותר מתווספות לאחר רשומות קודמות). כשאתה קובע את סדר המיון, עליך לציין את מספר העמודה שעליה הוא חל. מספור העמודות הוא מבוסס-אפס (מספר העמודה הראשונה הוא 0).

```
Sub AddIdx()

Dim cat1 As New ADOX.Catalog
Dim tbl1 As New ADOX.Table
Dim idx1 As New ADOX.Index

' Create a context for the new index.
    cat1.ActiveConnection = CurrentProject.Connection
    Set tbl1 = cat1.Tables("FamilyMembers2")

' Set the index properties.
    With idx1
        .Name = "LastIsFirst"
        .Unique = True
        .IndexNulls = adIndexNullsDisallow
    End With
```

```
' Append column to index and set its sort order.
' Append new index to table.
    idx1.Columns.Append "FamID"
    idx1.Columns(0).SortOrder = adSortDescending
    tbl1.Indexes.Append idx1

' Free resources.
    Set cat1 = Nothing
End Sub
```

אכלוס טבלה באופן דינמי

לאחר שעצבת טבלה והגדרת את האינדקסים שלה, עליך לאמץ אחת משתי גישות כדי למלא אותה בערכים. כשנתוני הטבלה המיועדת לאכלוס נמצאים בטבלה אחרת, באפשרותך להגדיר ערכת רשומות בטבלת מקור הנתונים ובטבלה החדשה. לאחר מכן עליך לנווט בשתי ערכות הרשומות בצורה מסונכרנת תוך העתקת רשומות לטבלה החדשה באמצעות השיטה AddNew (גישה זו עוברת בלולאה על ערכת רשומות, ולכן אינה פתרון מעשי לטבלאות גדולות מאוד). הגישה השנייה היא שימוש ב-SQL להוספת ערכים לטבלה החדשה, תוך התבססות על ערכי הטבלה המקורית. גישה זו אינה מבוססת על ערכות רשומות או השיטה AddNew, אך היא דורשת אובייקט Command אחד לפחות.

שימוש בערכות רשומות

הגישה הראשונה מומחשת באמצעות השיגרה הבאה, ודורשת אובייקט Connection ואובייקט Catalog נוסף לזוג ערכות רשומות (Recordsets). האובייקט Catalog ושתי ערכות הרשומות חולקים אובייקט Connection משותף. אובייקטים רבים מאוד חולקים ביניהם חיבור משותף, ולכן יש טעם להצהיר על אובייקט Connection ולהפעילו, לכל אחד מהאובייקטים האחרים הזקוקים לו.

```
Sub AddValues()
Dim cnn1 As ADODB.Connection
Dim cat1 As New ADOX.Catalog
Dim rst1 As New ADODB.Recordset
Dim rst2 As New ADODB.Recordset

' Set context for populating new table (FamilyMembers2).
    Set cnn1 = CurrentProject.Connection
    Set cat1.ActiveConnection = cnn1
    Set rst1.ActiveConnection = cnn1
'    Set rst2.ActiveConnection = cnn1
```

```
' Open recordsets based on new and original tables.
    rst1.Open "FamilyMembers2", , adOpenKeyset, _
        adLockOptimistic, adCmdTable
    rst2.Open "FamilyMembers", cnn1, adOpenForwardOnly, _
        adLockReadOnly, adCmdTable

' Loop through recordsets to copy from original to new table.
    With rst1
        Do Until rst2.EOF
            .AddNew
                .Fields(0) = rst2.Fields(0)
                .Fields(1) = rst2.Fields(1)
                .Fields(2) = rst2.Fields(2)
                .Fields(3) = rst2.Fields(3)
'            .Update
            .MoveNext
            rst2.MoveNext
        Loop
    End With
End Sub
```

לאחר ההצהרה על אובייקטים והקצאת החיבור לאובייקטים שישתמשו בו, השיגרה פותחת ערכות רשומות בטבלה המכילה את הרשומות המקוריות, FamilyMembers, ובטבלה החדשה, FamilyMembers2.

הקוד שלעיל מציג שתי גישות להקצאת חיבור לערכת רשומות. באפשרותך להקצות חיבור למאפיין ActiveConnection של ערכת רשומות – AddValues משתמשת בטכניקה זו בטיפול ב-rst1. בגישה השנייה, תוכל להפנות אל אובייקט Connection בשיטה Open של ערכת רשומות – השיטה Open הפועלת על rst2 משתמשת בטכניקה זו. ניתן לפעול בכל אחת משתי הגישות בעבודה עם שתי ערכות הרשומות. השיגרה AddValues כוללת שורת קוד שמראה כיצד להגדיר את המאפיין ActiveConnection של rst2 באמצעות פעולת הקצאה. שורה זו מסומנת בתור הערה. אם תחליט לנקוט גישה זו בטיפול ב-rst2, תוכל להסיר את ההפניה אל cnn1 בשיטה Open הפועלת על rst2.

המשפט Do...Loop העובר על הרשומות של rst2 מקונן בתוך המשפט With...End המתייחס אל rst1 (עקב הפניות הרבות אל rst1 בלולאה, משפט זה מאיץ את פעולת הקוד ומשפר את קריאותו). בתוך הלולאה, השיטה AddNew מאפשרת את הקצאת הרשומה הנוכחית ב-rst2 אל rst1. השימוש בשיטה Update להשלמת פעולת השיטה AddNew הוא אופציונלי (הדוגמה מציגה אותה בצורת הערה, ולא כחלק פעיל בקוד). לאחר הוספת הרשומה הנוכחית מ-rst2 ל-rst1, השיגרה מתקדמת לרשומה הבאה ב-rst2. אין צורך להתקדם גם ב-rst1, מכיון שמעבר הלולאה הבא יוסיף רשומה חדשה בסוף ערכת הרשומות rst1. הלולאה Do ממשיכה להתבצע עד הגיעה לסוף הקובץ (EOF) של rst2.

שמירת ערכת רשומות

לאחר שאיכלסת בערכים את ערכת הרשומות והטבלה שעליה היא מבוססת (ואולי אף עדכנת אותן), באפשרותך לשמור עותק של ערכת הרשומות. עושים זאת באמצעות שיטת ערכת הרשומות החדשה Save בהתאם לסוג הקובץ שלה, המבוסס על תבנית בינארית קומפקטית. Access 2000 מציע כמה דרכים לשמירה ואחזור של ערכות רשומות באמצעות תבנית חדשה זו והַסָפק המתאים שלה. השגרות שלפניך שומרות ערכת רשומות על ידי שימוש בסמן לקוח (client cursor), ולאחר מכן פותחות אותה.

```
Sub SaveRST()
On Error GoTo SaveRSTErr
Dim rst1 As New ADODB.Recordset
' Open client cursor and recordset.
    rst1.CursorLocation = adUseClient
    rst1.Open "FamilyMembers2", CurrentProject.Connection, _
        adOpenStatic, adLockBatchOptimistic
    rst1.Save "c:\FamilyMembers3.adtg", adPersistADTG
SaveRSTExit:
    Exit Sub
SaveRSTErr:
' Trap file already exists error
    If Err.Number = -2147286960 Then
        Kill "c:\FamilyMembers3.adtg"
        Resume
    End If
' Exit for other errors.
    MsgBox "View Immediate window for error diagnostics.", _
        vbCritical, "Programming Microsoft Access 2000"
    Resume SaveRSTExit
End Sub

Sub OpenSavedRST()
Dim rst1 As New ADODB.Recordset
' Open saved recordset file.
    rst1.Open "c:\FamilyMembers3.adtg", "Provider=MSPersist"
' Print selected info to confirm retrieval.
    Debug.Print rst1.Fields(0).Name & " = " & rst1.Fields(0).Value
End Sub
```

חלקה הראשי של השיגרה SaveRST פותח סמן לקוח ומציב בו ערכת רשומות. לאחר מכן הוא מפעיל שיטה Save של ערכת רשומות. שמירת ערכת רשומות לקובץ אינו תהליך מורכב יותר מכך. הקובץ ADTG (Advanced Data TableGram) הוא מינימלי בגודלו (1KB בלבד בגירסה של מחבר הספר). יש בכך חיסכון משמעותי לעומת אחסון ערכת רשומות בצורת טבלה בקובץ מסד נתונים.

אם אתה רגיל לשמור ערכת רשומות לאחר עדכונה, השיטה Save תיכשל אם לא תמחק את שם הקובץ הישן, או תיתן שם חדש. לחילופין, באפשרותך ללכוד את השגיאה, למחוק את גרסת הקובץ הישנה ולאחר מכן להפעיל שנית את השיטה Save. שגרת הטיפול בשגיאות של SaveRST מדגימה טכניקה זו.

באפשרותך לפתוח ערכת רשומות שמורה באמצעות שתי שורות קוד בלבד, כפי שמדגימה OpenSaveRST. השורה הראשונה מצהירה על אובייקט Recordset ויוצרת מופע שלו. השורה השנייה מפעילה את השיטה Open של ערכת הרשומות. המקור שלה הוא הקובץ ששמרה השיגרה SaveRST, וארגומנט החיבור שלה מפנה לספק MSPersist הכלול בחבילת המוצר Access 2000. שורה שלישית מדפיסה את תוכן השדה מתוך ערכת הרשומות כדי לאשר שהמידע שבקובץ תקין.

שימוש ב-SQL

רעיון המעבר בלולאה על ערכת רשומות, רשומה אחר רשומה בזמן נתון קוסם למפתחים רבים עקב מוחשיות הפעולה. אכן, קל לדמות את המעבר מרשומה לרשומה ב-rst2 ואת ההוספה של רשומות יחידות ל-rst1. יחד עם זאת, יעיל יותר לבצע את המשימה בעזרת קוד SQL. בנוסף, הגישה המבוססת על SQL זקוקה לאובייקט Command אחד בלבד מתוך הספריה ADODB. גישת המעבר בלולאה מסתמכת על אובייקטים אחדים הלקוחים מתוך הספריות ADODB ו-ADOX יחדיו. אם קשה לך לכתוב קוד SQL, באפשרותך לנצל את מעצב השאילתות החזותי של Access לקבלת גירסה ראשונית ותפקודית של תחביר SQL תקין לביצוע משימה, כגון הוספת רשומות מתוך טבלה אחת לטבלה אחרת. השיגרה AddValuesSQL ממחישה את יתרונותיו החסכוניים של קוד SQL (לאמיתו של דבר, אם היישום שלך אינו חייב לדעת את מספר הרשומות שהוא מוסיף, ניתן לקצר עוד יותר את הקוד).

```
Sub AddValuesSQL()
Dim cmd1 As ADODB.Command
Dim intRowsAdded As Integer
Dim str1
' Instantiate cmd1.
   Set cmd1 = New ADODB.Command
' Set the connection and text for cmd1
' before executing it.
   With cmd1
      .ActiveConnection = CurrentProject.Connection
      .CommandText = "INSERT INTO FamilyMembers2 " & _
         "SELECT FamilyMembers.* FROM FamilyMembers;"
      .CommandType = adCmdText
      .Execute intRowsAdded
   End With
'Report how many records cmd1 added.
   str1 = intRowsAdded & " rows were added to the table."
   MsgBox str1, vbInformation, "Programming Microsoft Access 2000"
End Sub
```

השיגרה פותחת בהצהרה ויצירת מופע של אובייקט Command. היא משתמשת במשפט With...End כדי לפשט את הגדרת מאפייני האובייקט ולהפעיל את השיטות שלו. קטע הקוד הכלול במשפט With...End מתחיל בקביעת המאפיינים ActiveConnection ו-CommandText של האובייקט Command (בגישה זו, טקסט הפקודה יהיה תמיד משפט SQL). קטע הקוד מגדיר את המאפיין CommandType בתור adCmdText כדי למטב את הביצועים (אם לא תגדיר את המאפיין CommandTyp, Access יצטרך לקבוע את הסוג של האובייקט Command בטרם יוכל לטפל בפקודה). לבסוף, קטע הקוד קורא לשיטה Execute עם הארגומנט האופציונלי intRowsAdded. בסיום השיגרה, intRowsAdded מכיל את מספר הרשומות שמשפט SQL טיפל בהן. משפט תיבת הודעה נוסף מנצל את intRowsAdded כדי לדווח למשתמש את מספר הרשומות שנוספו לטבלה החדשה.

עבודה עם נתונים בתבניות אחרות

בפרק 2 למדנו כיצד להשתמש בספק Jet 4 OLE DB כדי לפתוח את מסד הנתונים Northwind מתוך יישום. באפשרותך לעשות זאת כדי לגשת לנתונים שנמצאים במסד נתונים מסוים של Access מתוך כל מסד נתונים אחר של Access. שיטת עבודה זו ישימה אפילו לנתונים המאוחסנים בתבניות אחרות שעבורן קיים ספק OLE DB. גם במקרים שספק OLE DB אינו זמין, באפשרותך ליצור קישור אל מסד נתונים אחר באמצעות DAO, לגשת בקלות למידע שהוא מכיל ולנצל את יתרונותיו של מקור נתונים מקושר כלשהו.

באפשרותך לנצל את ספק Jet 4 כדי לפתוח מקור נתונים מסוג Excel. תרשים 3.15 מציג גיליון עבודה פשוט בחוברת עבודה של Excel, שנקראת Customers.xls. הגיליון Sheet1 מכיל טווח בשם Customers (ראה **תיבת שם** - Name Box) ובו מידע אודות קוד לקוח, שם החברה, שם איש קשר ותנאי תשלום. השיגרה שלפניך פותחת טווח בחוברת עבודה של Excel 2000 באמצעות Access 2000.

```
Sub OpenPrintXLDataSource()
Dim cnn1 As New ADODB.Connection
Dim rst1 As Recordset
' Open and set recordset.
   cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
       "Data Source=C:\Programming Access\Chap03\Customers.xls;" & _
       "Extended Properties=Excel 8.0;"
   Set rst1 = New ADODB.Recordset
   rst1.CursorType = adOpenKeyset
   rst1.LockType = adLockOptimistic
   rst1.Open "customers", cnn1, , , adCmdTable
' Open recordset and print a test record.
   Do Until rst1.EOF
       Debug.Print rst1.Fields(0).Value, rst1.Fields(1).Value
       rst1.MoveNext
   Loop
End Sub
```

Microsoft Excel - Customers

File Edit View Insert Format Tools Data Window Help

Customers = CustomerID

	A	B	C	D	E	F
1	CustomerID	CompanyName	ContactFirstName	ContactLastName	PaymentTerms	
2	VBD	Visual Basic Developer	Lori	Coffey	Day of publication	
3	MOD	Microsoft Office & VBA Developer	Kaye	Waterhouse	30 days after publication	
4	IW	Internet World	Bob	Danners	Day of seminar	
5	SA	Smart Access	Paul	Augustus	Upon approval	
6	BYTE	Byte	Tom	Davis	Upon approval	
7	WT	Web Techniques	Dave	DeMott	When commissioned	
8	MODJ	Microsoft Office Developer's Journal	Tami	Dreyfuss	Day of publication	
9	VBPJ	Visual Basic Programmer's Journal	Jim	Donaldson	End of Month for publication	
10	AOVBA	Access Office VB Advisor	Mike	Giddy	30 days after publication	
11	MWB	Microsoft Web Builder	Sunil	Mehrotra	Day of publication	
		Active Server Developer's			Day of	

Sheet1 / Sheet2 / Sheet3

Ready

תרשים 3.15: Access יכול לטפל במידע שבגיליון עבודה של Excel

השיגרה OpenPrintXLDataSource משתמשת באובייקט Connection כדי לפתוח חוברת עבודה של Excel. השיטה Open של האובייקט cnn1 יוצרת חיבור אל קובץ חוברת העבודה. הספק בקריאה זו הוא אותו ספק בו השתמשנו עבור קבצי Jet 4, הארגומנט Extended Properties מציין תבנית תואמת לקבצי Excel 2000 וארגומנט מקור הנתונים מצביע על חוברת עבודה של Excel בספריה Programming Access שבכונן C. ארבע השורות הבאות בשיגרה יוצרות ופותחות הפניה לערכת רשומות שמוגדרת בטווח Customers בקובץ Customers.xls. לולאת Do שבסוף השיגרה, מדפיסה את שני השדות הראשונים שבערכת הרשומות לחלון Immediate.

לאחר פתיחת ערכת רשומות עבור מקור נתונים ב-Excel, באפשרותך לבצע כל פעולה ש-Access מאפשר לבצע על מקור רשומות, כגון עדכון ערכי שדות. ההבדל במקרה זה, הוא שהיישום מעדכן ערכים בטווח בחוברת עבודה של Excel, ולא ערכים בטבלה של Access . שגרת הדוגמה OpenPrintXLDataSource ממחישה בצורה ברורה את עוצמתם של ספקי ADO ו-OLEDB כחלק מפעולת יישום Access.

שימוש בספק MSDASQL

Access 2000 עובד עם ספקי OLEDB נוספים, כגון MSDASQL, הפועל על כל מקורות הנתונים מסוג ODBC. למרות שקיים ספק מיוחד עבור Microsoft SQL Server, השיגרה שלפניך מנצלת את הספק הכללי ODBC OLEDB כדי להמחיש את השימוש בספק בעבודה עם מקור נתונים ODBC כלשהו. ספק זה מתייחס תמיד למקור נתונים ODBC, ולכן באפשרותך להשתמש בו בתיאום עם DSN. השיגרה מציגה ספק MSDASQL המפנה למסד נתונים SQL Server Pubs, כדי להדפיס את פרטי המחבר: מספר זהות, שם פרטי ושם משפחה.

```
Sub GetODBCThroughOLEDB()
Dim cnn1 As New ADODB.Connection
Dim rst1 As ADODB.Recordset
' Open ODBC sources with msdaSQL provider and DSN reference.
   cnn1.Open "Provider=MSDASQL;DSN=Pubs;"
   Set rst1 = New ADODB.Recordset
   rst1.CursorType = adOpenKeyset
   rst1.LockType = adLockOptimistic
   rst1.Open "authors", cnn1, , , adCmdTable
' Open recordset and print a test record.
   Do Until rst1.EOF
      Debug.Print rst1.Fields(0).Value, rst1.Fields(2), rst1.Fields(1).Value
      rst1.MoveNext
   Loop
End Sub
```

השיגרה GetODBCThroughOLEDB מתחילה בפתיחת חיבור אל מסד הנתונים SQL Server Pubs ומפעילה את השיטה Open של האובייקט. ארגומנט הספק מציין את ספק MSDASQL, ואילו הארגומנט DSN מכיל את נתוני החיבור ופועל בתור ארגומנט Data Source של ספק Jet 4. השיטה Open יכולה לציין את נתוני מחרוזת החיבור, מה שמבטל את הצורך להתייחס ל-DSN. המשפט שלפניך מראה כיצד פותחים מסד נתונים SQL Server Pubs מבלי לציין DSN:

```
cnn1.Open "Provider=MSDASQL;DRIVER=SQL Server;" & _
   "SERVER=CAB2200;DATABASE=Pubs;uid=sa;pwd=;"
```

שמו של SQL Server הוא CAB2200. שים לב שמחרוזת החיבור מכילה גם את הארגומנטים של זיהוי המשתמש והסיסמה. מחרוזת החיבור ייחודית למסד הנתונים שאליו אתה מתחבר.

ארבע השורות הבאות של השיגרה פותחות ערכת רשומות בטבלה Authors במסד הנתונים SQL Server Pubs. תחביר השיטה Open זהה לזה ששימש לפתיחת טווח בחוברת העבודה של Excel. היחס הסימטרי בין שני מקורות שונים אלה ממחיש את עוצמתו של ADO עם ספקי OLE DB. לולאת Do הסופית עוברת על הרשומות כדי

להדפיס את מספר הזהות, השם הפרטי ושם המשפחה של כל מחבר. המאפיין Value הוא ברירת המחדל של האוסף Fields של האובייקט ערכת רשומות, ולכן אין צורך לציין אותו (שים לב שהארגומנט השני של השיטה Debug.Print אינו מפנה מפורשות למאפיין Value של האוסף Fields). העיצוב הכולל של השיגרה GetODBCThroughOLEDB מראה ש- Access 2000 יכול ליצור קישור אל מסד נתונים SQL Server באותה קלות שהוא יוצר קישור אל חוברת עבודה של Excel.

השיטות TransferDatabase ו-TransferSpreadsheet של האובייקט DoCmd הן אמצעי קל וחסין תקלות לקישור לנתונים שנמצאים ביישום אחר; הן אף אינן תלויות בהפניה לספריה DAO. שיטות אלו חושפות את מקורותיהן באמצעות החלון **מסד נתונים**, בעוד שהספק OLE DB מספק ממשק תכנותי בלעדי למקורות הנתונים שלו. יכולתך להפיק תועלת מחשיפת המקור באמצעות החלון **מסד נתונים** תלויה ביישום שלך ובהעדפות של המשתמשים בו.

שימוש בשיטה TransferDatabase או בשיטה TransferSpreadsheet

השיטות TransferDatabase ו-TransferSpreadsheet הן תוצאה של פקודות מאקרו המאפשרות ייבוא, קישור וייצוא. השיטה TransferSpreadsheet תומכת במיגוון רחב של תבניות Lotus ו-Excel. השיטה TransferDatabase תומכת במקורות נתונים ODBC, כגון SQL Server ו-Oracle, וגם במקורות נתונים ISAM, כגון Paradox, dBase, Microsoft FoxPro ו-Jet. השיטות TransferDatabase ו-TransferSpreadsheet מנצלות את מנהלי ההתקנים של ISAM ו-ODBC המשווקים עם Access. מידע רב-ערך אודות שיטות אלו ניתן למצוא במערכת העזרה המקוונת של Access ובחלון **מאקרו** הפתיח בתצוגת **עיצוב**, הכולל תיבות רשימה נפתחת שמהן ניתן לבחור ארגומנטים אפשריים.

השיגרה linkXLCustomers שלפניך מציגה את תחביר השיטה TransferSpreadsheet. באפשרותך להשוות טכניקה זו עם התפקודיות של ספק OLE DB. שגרת הדוגמה יוצרת קישור לטווח Customers בחוברת העבודה Customers.xls (הקובץ Customers.xls כלול בתקליטור המצורף לספר). חוברת העבודה מבוססת על תבנית Excel 97, אך תומכת גם בגרסאות 3-5 של Excel. TransferSpreadsheet תומכת גם בתבניות של Lotus, כגון WK1, WK3 ו-WK4. האובייקט DoCmd יוצר טבלה מקושרת שנקראת XLCustomers במסד הנתונים. הארגומנט האחד לפני אחרון של השיטה TransferSpreadsheet (‎-1) מציין כי הטווח של Excel כולל שמות שדות בשורה הראשונה.

```
Sub linkXLCustomers()
Dim cnn1 As New ADODB.Connection
Dim rst1 As ADODB.Recordset
' Use DoCmd to programmatically make the link.
   DoCmd.TransferSpreadsheet acLink, acSpreadsheetTypeExcel97, _
   "XLCustomers", "C:\Programming Access\Chap03\Customers.xls", _
   -1, "Customers"
```

```
' Open and set recordset.
    Set rst1 = New ADODB.Recordset
    rst1.ActiveConnection = CurrentProject.Connection
    rst1.CursorType = adOpenKeyset
    rst1.LockType = adLockOptimistic
    rst1.Open "XLCustomers", , , , adCmdTable
' Open recordset and print a test record.
    Do Until rst1.EOF
        Debug.Print rst1.Fields(0).Value, rst1.Fields(2)
        rst1.MoveNext
    Loop
End Sub
```

לאחר שהשיטה TransferSpreadsheet מתבצעת, היישום יכול לנצל את קבוצת ההחזרה (return set) שלו בצורה דומה לזו שמתקבלת מספק OLE BD. ראשית, פתח את ערכת הרשומות בטבלה המקושרת. לאחר מכן טפל בערכת הרשומות בעזרת קוד כדי למלא את דרישות היישום (LinkXLCustomers מדפיסה שדות אחדים לכל שורה).

השיטה TransferDatabase כוללת מיגוון גדול יותר של אפשרויות גישה לנתונים לעומת הפקודה TransferSpreadsheet. האפשרויות כוללות את מנהלי ההתקנים ההיסטוריים של ISAM ו-ODBC. הואיל ורוב מסדי הנתונים תומכים ב-ODBC, שיטה זו יכולה לגשר כמעט אל כל מקור נתונים של מערכת לניהול מסדי נתונים יחסיים (RDBMS). יתירה מזאת, התחביר של השיטה TransferDatabase דומה לזה של TransferSpreadsheet, מה שמקל את השימוש בה.

השיגרה LinkODBCAuthors מנצלת את השיטה TransferSpreadsheet כדי ליצור קישור אל הטבלה Authors במסד הנתונים Pubs של SQL Server. LinkODBCAuthor יוצרת קישור אל הטבלה Authors תחת השם adoAuthors במסד הנתונים הנוכחי. אם adoAuthors כבר קיימת במסד הנתונים הנוכחי, השיטה שומרת את העותק המקורי מבלי להתריע על כך, יוצרת עותק נוסף ונותנת לו את השם adoAuthors1.

```
Sub linkODBCAuthors()
Dim cnn1 As New ADODB.Connection
Dim rst1 As ADODB.Recordset
' Use DoCmd to programmatically make the link.
    DoCmd.TransferDatabase acLink, "ODBC Database", _
        "ODBC;DSN=Pubs;UID=sa;PWD=;DATABASE=pubs", _
        acTable, "Authors", "dboAuthors"
' Open and set recordset.
    Set rst1 = New ADODB.Recordset
    rst1.ActiveConnection = CurrentProject.Connection
    rst1.CursorType = adOpenKeyset
    rst1.LockType = adLockOptimistic
    rst1.Open "dboAuthors", , , , adCmdTable
```

```
' Open recordset and print a test record.
    Do Until rst1.EOF
        Debug.Print rst1.Fields(0).Value, rst1.Fields(2)
        rst1.MoveNext
    Loop
End Sub
```

הארגומנט הראשון המועבר אל TransferDatabase מציין את סוג ההעברה אל מקור הנתונים; הארגומנט acLink יוצר טבלה מקושרת במסד הנתונים. שני הארגומנטים הבאים מציינים את סוג מסד הנתונים ואת שמו. (השיגרה LinkODBCAuthors מגדירה מקור נתונים מסוג ODBC. לאמיתו של דבר, דוגמה זו משתמשת במסד הנתונים המקובל Pubs של SQL Server). הארגומנטים הנותרים מציינים את סוג אובייקט מסד הנתונים שעליו ברצונך לבסס את המקור, את שמו במקור המרוחק (Authors) ואת שמו ביישום שלך (dboAuthors). שאר חלקי הקוד של השיגרה מבצעים פעולות זהות לאלו של שאר דוגמאות הקוד בסעיף זה.

4

טיפול בנתונים באמצעות שאילתות

שאילתות הן סוס העבודה של יישומי מסד נתונים. תוכל ליישם אותן כדי להשיג מטרות רבות. שאילתות מאפשרות לטפל בנתונים שנמצאים בטבלאות מסד נתונים. תוכל לנצלן כדי להגדיר תכנים של טפסים ודוחות. שאילתות יכולות גם לציין את מקור הנתונים של דף Web.

אחת הסיבות העיקריות לכוחן הרב של שאילתות היא המיגוון העצום שלהן. בפרק 2 למדנו כיצד לעצב שאילתות החזרת שורות (row-returning) ושאילתות bulk operation (שאילתות פעולה בתבנית המוגדרת על ידי המשתמש), באמצעות אובייקטי נתונים ActiveX (ADO). ניתן גם לבצע פעולות אלה ואחרות באמצעות שפת שאילתה מובנית (SQL). מפתחים משתמשים לעיתים קרובות ב-SQL כדי לטפל בטבלה על ידי חילוץ שורות ועמודות נבחרות, אך אלו אינן האפשרויות היחידות לטיפול במקור נתונים. שאילתות bulk operation קלאסיות כוללות צירוף ומחיקה של רשומות ועדכון שדות. ניתן גם ליצור שאילתות חדשות באמצעות SQL.

בפרק זה נדון בנושא השאילתות של Access 2000 במיגוון רמות.

- סקירה כללית על סוגי השאילתות השונות. הפרק עוסק בטיבן של שאילתות, במה שהן מאפשרות לעשות ובאופן בו הן משלימות זו את זו.
- תיאור טכניקות ידניות לבניית שאילתות. סקירה זו משרתת מטרות רבות. היא מרחיבה את הסקירה הכללית בהצגת שלבים ביצירת כל אחד מסוגי השאילתות.
- הפיכת משימה לאוטומטית באמצעות שאילתת פעולה. אפשר לנצל את VBA כדי ליישם באמצעות קוד שאילתות החזרת-שורות ושאילתות פעולה פשוטות.
- התבססות על SQL באמצעות קוד. כאן נשלים את הסקירה מפרק 2 בנושאי ADO ונדון בתצוגות ושגרות מאוחסנות.
- טיפול במסדי נתונים מרוחקים. גרסת 2000 מטפלת במסדי נתונים מרוחקים טוב יותר מגרסאות קודמות. שימוש בשאילתות למטרה זו ישפר את היישומים שתכתוב ויעלה את רמת הביצועים שלך בתור מפתח בסביבת Access.

סקירת סוגי שאילתות

ניתן להתייחס לשאילתות ככלים לחילוץ מידע ממסד נתונים אחד או יותר, אך קיים מיגוון עצום של דרכים בהן שאילתות מבצעות משימה זו. בנוסף, שאילתות יכולות לבצע פונקציות מסוגים שונים, במיוחד אם כוללים את תכונת הגדרת הנתונים של SQL וקישור למקורות נתונים מרוחקים. בסעיף זה נסקור אפשרויות שאילתה רבות ש- Access 2000 תומך בהן. סקירה זו תספק את המידע הדרוש כדי לבחור בסוג השאילתה הנכון למצב נתון.

שאילתות בחירה (Select)

שאילתות בחירה מאפשרות ליצור קבוצת משנה של פריטי מידע בטבלת מסד נתונים אחת או יותר. לרשות המפתחים עומד מערך אפשרויות עשיר שמאפשר בחירה ועיבוד נתונים בדרכים שונות. אפשרויות אלו יסייעו לך לקבוע את ערכי העמודה והשדה בקבוצה המוחזרת (Return Set) מתוך שאילתה. שאילתות בחירה הן מקור רשומות טוב עבור טפסים, דוחות ודפי Web, ולכן שליטה בעקרונות הבסיסיים שלהן תשפר מאוד את עבודתך עם רכיבי פיתוח נוספים.

שאילתה הבוחרת קבוצת משנה של שדות ברשומה היא אחת השאילתות הבסיסיות ביותר שיש. נוח מאוד להשתמש בה כשרוצים להציג עמודות אחדות מתוך טבלה הכוללת עמודות רבות. חילוץ מספר עמודות חלקי יכול להאיץ את פעולת השאילתה. Access מחשב שאילתת בחירה בכל פעם שהוא פועל, מובטח כי המשתמש יקבל תמיד את המידע העדכני ביותר. כשיישום אינו אמור לחשוף את המידע העדכני ביותר מהטבלה, יש להשתמש בקריטריונים שונים כדי להגביל את ערכי השורה שהשאילתה מחזירה. לחילופין, תוכל להפעיל סוג אחר של שאילתה כדי ליצור טבלה זמנית חדשה שמכילה קבוצת משנה של עמודות ברגע מוגדר בציר הזמן.

קביעת קריטריונים

בין אם שאילתת בחירה בוחרת קבוצת משנה מתוך מקור נתונים ובין אם לאו, היא יכולה להגביל את השורות בקבוצת המשנה. באפשרותך לציין את השורות שתוחזרנה על ידי קביעת קריטריונים מתאימים. השתמש בשורה **קריטריונים** (Criteria) ברשת עיצוב שאילתה כדי לקבוע קריטריונים. קביעת קריטריונים באמצעות קוד כוללת את הפסוקיות WHERE ו-HAVING במשפטי SQL. הפסוקית WHERE נוגעת לרשומות בודדות במקור הרשומות עליו מבצעים את השאילתה, אך הפסוקית HAVING יכולה להגביל את קבוצות ההחזרה בהתאם לערכי הפסוקית GROUP BY.

שאילתות בחירה יכולות להחזיר קבוצות שורות שמתאימות לערך או לטווח ערכים מסוים. בעת טיפול בטבלת ערכים גיאוגרפיים, ניתן להגדיר שאילתת בחירה שתחזיר את שם העיר והמדינה (בארה״ב) בהתאם למיקוד. אם הרשומות במקור נתונים מכילות שדה מסוג **תאריך/שעה**, כגון תאריך הזמנה או חשבונית, תוכל לבחור הזמנות או חשבוניות מתאריך מוגדר. ניתן להשתמש גם באופרטורי השוואה, כגון גדול (>),

קטן (<) או שונה (<>), כדי לבחור טווח שורות. אפשר להתאים את הפריט המבוקש לערך מחרוזת מוגדר, או לטווח ערכים באמצעות תווים כלליים (wild card). תווים אלה משמשים לקביעת טווח ערכי מחרוזת. לדוגמה, הקריטריון S*l מתאים למחרוזות Sal ו-Saul, מכיון שהתו * מתאים למספר תווים כלשהו. לעומת זאת, התו ? מתאים לתו אחד בלבד, ולכן הקריטריון S?l מתאים למחרוזת Sal, אך לא למחרוזת Saul.

פונקציות צבירה (aggregate)

מפתחים יכולים להשתמש בפונקציות צבירה (Aggregate Functions) בשילוב SQL, כדי לסכם את הרשומות במקור נתונים. פונקציות צבירה טיפוסיות מונות, מסכמות או ממצעות את רשומות מקור הנתונים כתגובה לשאילתת בחירה. ניתן להפעיל פונקציות צבירה כדי לחשב את מידת ההשתנות (Variability) של מקור רשומה (record source) באמצעות פונקציות כגון stDev או VarP. אפשר אף לחשב את המינימום או את המקסימום של קבוצת ערכים בשדה בעזרת פונקציות צבירה.

שאילתות בחירה יכולות ליישם פונקציות צבירה בשילוב עם הפסוקית GROUP BY. הדבר מאפשר לשאילתה לחשב נתוני צבירה עבור רשומות מקובצות במקור נתונים. שימוש בפונקציות צבירה יחד עם פסוקית GROUP BY מאפשר ליישום להחזיר את מספר שורות הפריט עבור כל הזמנה בשאילתת בחירה.

צירופים (joins)

אפשר לבסס שאילתות בחירה על טבלה אחת או יותר, על שאילתה אחרת או על שילוב כלשהו של טבלאות ושאילתות בחירה. בעת בחירה מתוך טבלה אחת או יותר, היישום ייקשר את הטבלאות באמצעות שדות משותפים אחדים. אם מסד הנתונים מקשר טבלאות באמצעות הגדרות שלמות הקשרים, Access יזהה זאת בעצמו ויצרף אותן כשתשלב אותן יחד בתור מקור רשומה של שאילתה. Access מזהה גם שדות משותפים במצבים נוספים, ועל ידי כך מאפשר לקיים, לשנות או להחליף קשרי גומלין שזוהו אוטומטית, בין טבלאות.

ברירת המחדל לצירוף טבלאות היא שילוב כל הרשומות בין שתי טבלאות שאחד משדותיהן מכיל ערך תואם. צירוף הטבלאות Orders ו- Orders Details של מסד הנתונים Northwind בשאילתת בחירה פשוטה, מסוגל להחזיר את כל שורות הפריטים לכל הזמנה. במצב זה אנו מעוניינים רק בפריטים שתואמים להזמנות. טבלאות הקשורות זו לזו יכולות להכיל בטבלה אחת ערכי שדות ללא ערכים תואמים בטבלה המצורפת (לדוגמה, תוכל לאחזר נתונים ממסד נתונים ישן שאינו מאלץ שלמות קשרים). היישום יכול לזהות רשומות שאין להן רשומות תואמות, על ידי הכללה מאולצת של כל רשומות טבלה מצורפת אחת או יותר לקבוצת החזרה. במצב זה, כל הרשומות המשולבות המכילות ערך מטבלה אחת, והחסרות שדה צירוף מהטבלה האחרת, תצבענה על רשומות לא תואמות.

תוצאת זוג כלשהו של ערכות רשומות מצורפות יכולה להצטרף לטבלה או לשאילתה אחרת. אם אתה מנוסה ב-SQL תוכל לפתח קבוצת החזרה על ידי טיפול במספר טבלאות ושאילתות באמצעות קשרי גומלין המבוססים על צירוף מורכב, זאת במשפט SQL אחד בלבד. מתלמד ב-SQL עשוי למצוא שבטוח יותר להעדיף בניית שאילתות בצורה גרפית ברשת עיצוב השאילתה כשהמדובר בזוגות טבלאות או שאילתות אחרות. כשמתכנני מסדי נתונים נוקטים צעד מקדמי זה, הם יכולים לאמת את תוצאת צירוף שתי הטבלאות לפני צירוף קבוצת ההחזרה לטבלה או שאילתה אחרת.

עדכון נתוני מקור

אחת התוצאות הנפוצות של שאילתת בחירה היא **קבוצת רשומות דינמית** (dynaset). קבוצת רשומות דינמית היא ערכת רשומות ששומרת מפתחות ראשיים במקום נתונים בפועל. שאילתת בחירה מאפשרת ליישום לשנות את השדות ברשומות שעליהן מבוצעת השאילתה (לשם הנוחות, נשתמש במילה רקע). לעיתים, השדות שברקע שאילתת בחירה אינם ניתנים לעדכון. בשאילתות בחירה המבוססות על טבלה בודדת או שתיים בקשר יחיד-ליחיד, קיימת תמיד אפשרות לעדכן את רשומות הרקע (אלא אם משתמש אחר נועל אותן). בדרך כלל ניתן לעדכן טבלאות שקשרי הגומלין שלהן הוא מסוג יחיד-לרבים.

הצג את השאילתה בתצוגת **גיליון נתונים** והבט בשורת המצב כדי לקבוע אם שדה מסוים הוא בר-עדכון. אם השדה אינו כזה ויש לך צורך לשנותו, שקול גישה ישירה לטבלה שבה הוא נמצא, או עיצוב מחדש של השאילתה.

שמירת שאילתות

כששאילתות הן מורכבות. כדאי לשמור אותן ואחר כך להפנות אליהן בשאילתות אחרות. לדוגמה, שאילתת בחירה המצרפת שתי שאילתות נוספות (או יותר), ולאחר מכן מפעילה עליהן פונקציות צבירה של SQL כדי ליצור סיכומי נתונים. שמור שאילתה מורכבת והפנה אליה בשאילתות אחרות כדי להימנע משגיאות תחביר. לעיתים, כשמצרפים שאילתה אחת או יותר כדי לקבל שאילתה חדשה, לוגיקת השאילתה עלולה להיות מורכבת מדי לכוח החישוב של מנגנון מסד הנתונים. במצב זה, שמור שאילתת קלט אחת או יותר כטבלה זמנית. לאחר מכן, השתמש בטבלאות אלו כתחליפים עבור השאילתות. חשב מחדש את הטבלאות הזמניות בכל מקרה שחל שינוי בנתוני הקלט של הטבלאות המקוריות.

שאילתות פעולה (Action)

במקום להחזיר קבוצת שורות כמו בשאילתת בחירה, **שאילתות פעולה** (Action Queries) מבצעות את משימתן מול טבלה אחת או יותר. שאילתות פעולה מאפשרות להפעיל אוטומציה של יישום. אפשר לעצב שאילתות פעולה באופן גרפי, והן תמשכנה לבצע את הפעולות החיוניות: הוספה, מחיקה ועדכון רשומות. שאילתות פעולה משתמשות ב-SQL לביצוע משימותיהן, ולכן ניתן לנצל אותן כפקודות טבלה

מתוחכמות לטיפול ברשומות. מפתחי Access מנוסים תוהים מן הסתם על שאילתת הפעולה של Access שנקראת שאילתת יצירת טבלה (make table). שאילתת פעולה זו מאפשרת הגדרת נתונים וגם טיפול בהם. שאילתת יצירת טבלה היא הנושא המרכזי בסעיף שאילתות יצירת טבלה שבהמשך הפרק.

שאילתות הוספה

שאילתות הוספה (Append Queries) מאפשרות להוסיף רשומות לטבלה אחת המבוססות על אלו שבטבלה אחרת. הפעל שאילתת הוספה כשהיישום מקבל רשומות ממקור חיצוני, כגון טבלה שנמצאת במחשב אחר. טבלת המקור חייבת להיות במסד הנתונים הנוכחי, או לפחות מקושרת אליו. טבלת היעד יכולה להיות במסד הנתונים הנוכחי או בקובץ מסד נתונים אחר. שמות השדות בשתי הטבלאות יכולים להיות שונים זה מזה, אך חייבת להיות תאימות בין סוגי הנתונים כדי לצרף ערכים משדה בטבלה אחת לשדה תואם בטבלה אחרת. טבלת המקור אינה מחייבת שכל השדות יהיו בטבלת היעד.

קיימים מצבים העלולים ליצור שגיאות בשאילתות הוספה. בעיקרון, עליך להימנע מהוספת ערכים לשדה מספור אוטומטי, מכיון ששדה מסוג זה מתאכלס מעצמו. אם תנסה להוסיף רשומה שגורמת לכפילות בשדה המפתח הראשי, היישום ייצור שגיאה של הפרת מפתח. Access לא יאפשר להוסיף רשומות, אם הוספתן מפירה את תקפות הטבלה או את כללי שלמות הקשרים.

שאילתות מחיקה

שאילתות מחיקה (Delete queries) מאפשרות ליישום להפוך את תהליך הסרת הרשומות מטבלה לתהליך אוטומטי. ניתן לקבוע את הרשומות המיועדות למחיקה בעזרת קריטריונים מוגדרים. אפשר למחוק שתי רשומות או יותר באמצעות שאילתת מחיקה בודדת. כאשר סכימת מסד הנתונים מציינת מחיקות על פי היררכית הקשרים, שאילתת מחיקה תמחק רשומות תואמות הן בצד היחיד והן בצד הרבים, בקשר גומלין מסוג יחיד-לרבים.

אם עליך למחוק רשומות מתוך צד היחיד בלבד, בטל את המחיקות על פי היררכית הקשרים מתוך הגדרות שלמות הקשרים. אם עליך לבטל את כל הרשומות בצד הרבים של קשר גומלין מסוג יחיד-לרבים בעל שלמות קשרים, כבה את קשר הגומלין, בצע את שאילתת המחיקה ואחר כך, שחזר את הגדרות שלמות הקשרים.

שאילתות עדכון

בניגוד לשאילתות הוספה ומחיקה, **שאילתות עדכון** (Update queries) אינן פועלת על רשומות שלמות. במקום זאת הן משנות ערכים בשדות שנבחרו. הקצה שורות מוגדרות באמצעות קריטריוני הגדרה. אפשר לנצל שאילתה מסוג זה לחישוב מחירי פריטים חדשים שעולים בשיעור קבוע. אם העליה במחירים חלה על קטגוריית מוצר אחת בלבד, ציין את הקטגוריה בהגדרת הקריטריונים.

סוגי שאילתה אחרים

קיימת קבוצת שאילתות מיוחדת נוספת שמרחיבה את שימושיות השאילתות. בדרך כלל, אם כי לא תמיד, שאילתות אלו מרחיבות או משלימות את היכולת של שאילתות בחירה והוספה.

שאילתות פרמטר

שאילתות אלו הן סוג מיוחד של שאילתות שיכולות להחזיר שורות או לבצע פעולות. בזמן ריצה, שאילתה מסוימת יכולה להנחות את המשתמש לספק קלט שקובע את אופן פעולתה. באפשרותך לבקש מהמשתמש קלט אחד או יותר על ידי שימוש בסוגי נתונים שונים. מורים לשאילתת הפרמטר מה עליה לבצע על ידי הכנסת ערכים בתשובה לבקשות שלה, או על ידי הגדרת הפרמטרים שלה באמצעות קוד VBA טרם ביצוע השאילתה, כדי לשלוט בקבוצת ההחזרה שלה או בפעולה שהיא מבצעת. הדבר מאפשר לספק קוד זיהוי של לקוח בזמן ריצה, כדי לקבוע את הלקוח ששאילתת הבחירה תחזיר את פרטיו.

כחלופה לשאילתת פרמטר, היישום יכול להפנות למחרוזת SQL של שאילתת בחירה או פעולה, באמצעות משתני מחרוזת. בטרם תבצע את משפט SQL בפקודת ADO, הקצה ערכים מוגדרים למשתני המחרוזת. על ידי כך ניתן לקבל תוצאות גמישות יותר מאשר מקבלים בשאילתת פרמטר, מכיון באפשרותך לשנות, למעשה, פסוקיות שלמות במשפט SQL של שאילתה. במקרים מסוימים, שאילתות פרמטר מפצות על כך באמצעות הקלדת נתונים ובקשות מוכללות לקבלת ערכים. בנוסף, שאילתות פרמטר מבטלות את הצורך במניפולציות של מחרוזות SQL, כל שעלינו לעשות הוא לעדכן את ערך הפרמטר.

שאילתת איחוד

שאילתת האיחוד (union query) היא סוג של שאילתת SQL. זוהי שאילתה מעניינת במיוחד, עקב התנהגותה החדשנית (היא מציבה טבלאות זו מאחורי זו במקום זו לצד זו). שאילתות איחוד מצרפות יחד את השדות מתוך שתי טבלאות או יותר. במקום לצרף את רשומות הטבלאות זו לצד זו, שאילתות איחוד מוסיפות את הרשומות מטבלה אחת ישירות אחרי אלו של הטבלה האחרת. עליך לבנות שאילתות איחוד על ידי שימוש במשפטי SQL. לא ניתן לעצב אותן בצורה גרפית ברשת עיצוב השאילתה. הקלד את קוד SQL של שאילתת האיחוד ישירות בתצוגת SQL של השאילתה. תכונה זו מייחדת את שאילתות האיחוד מכל סוגי השאילתות בהן עסקנו עד כה, מכיון שבאפשרותך ליצור את כל שאר השאילתות בצורה גרפית. שאילתות איחוד מעניקות לך את מלוא העוצמה של שפת SQL להגדרת קריטריונים ולקביעת סדרי מיון.

שאילתות הצלבות

שאילתת הצלבות (crosstab query) קולטות טבלה או שאילתה בתור מקור נתונים ומחזירה סכום, ממוצע או מונה של שדה אחד המבוסס על שני שדות קטגוריה נוספים. שאילתה מסוג זה זקוקה לשדה שהיישום יכול למנות, לסכם או למצע. נוסף לציון שדה המיועד להצגת תוצאות צבירה, מפתחים חייבים לספק שדות קטגוריה של שורה ועמודה. שדות אלה צריכים להכיל ערכים בדידים שישמשו בתור קטגוריות לדיווח על תוצאות צבירה. Access תומך ביצירת שאילתות הצלבות באמצעות אשף ו-Jet SQL. בנוסף, באפשרותך לטפל ידנית בעיצוב שאילתת הצלבות בתצוגות **עיצוב** (Design) **וגיליון נתונים** (Datasheet).

שאילתות משנה

שאילתת משנה SQL היא משפט SELECT בשפת SQL שמקונן בתוך שאילתת פעולה אחרת. השתמש במשפט SELECT המקונן בתור ביטוי בקריטריון של שדה. המשפט המקונן מחזיר ערך שבבוא הזמן יכול לתפקד בתור קריטריון נוסף. למרות שעליך לכתוב את שאילתת המשנה בשפת SQL, תוכל לקנן אותה ברשת עיצוב השאילתה. יתירה מזאת – באפשרותך להשתמש בשאילתה נוספת כדי לאשר שעיצבת את המשפט המקונן בצורה תקינה. השתמש במשפט SQL המקונן כפי שהיית נוהג בקריטריון כלשהו אחר, כדי לציין ערך או תחום ערכים עבור קבוצת החזרה.

פעולות להגדרת נתונים

שפת SQL Data Definition Language (SQL DDL) מאפשרת למפתחים ליצור טבלאות המכילות משפטי SQL. השאילתה ליצירת טבלה מנצלת את DDL כדי להפוך יצירת טבלה לתהליך אוטומטי. אחד ההיבטים המצודדים של שאילתת יצירת טבלה הוא האפשרות לעצב אותה בצורה גרפית. מקובל מאוד להפעיל את עיצוב שאילתת יצירת טבלה בתור שאילתת בחירה סטנדרטית. לאחר עיצוב שאילתת בחירה כך שתחזיר בדיוק את מה שאתה מחפש, הפוך אותה לשאילתת יצירת טבלה.

השתמש ב- SQL DDL לעיצוב ידני של טבלאות באמצעות משפטי SQL. אם נוח לך לעבוד עם משפטי SQL, זוהי הדרך המהירה והטובה ביותר לעצב טבלאות. SQL DDL מהווה חלופה ל-ADO. קיימות משימות נבחרות, כגון קביעת ערך הצעד והערך ההתחלתי של סוג הנתונים **מספור אוטומטי** (AutoNumber), או הפעלה וביטול דחיסת Unicode, שניתן לקבל בלעדית ב- Jet SQL. כמובן, מפתחים לסביבת Access נוהגים להשתמש במשפטי SQL בעיקר כדי להגדיר את המאפיין CommandType של אובייקט Command של ADO. אם המאפיין CommandType בתוכנית שלך קיבל את הערך adCmdText, עליך להשתמש במשפט SQL כדי להגדיר את המאפיין CommandText. הפעל את השיטה Execute של הפקודה כדי להפעיל את משפט SQL.

עבודה עם מקורות נתונים מרוחקים

כשעובדים עם מקורות נתונים מרוחקים, כגון Microsoft SQL Server או מסד נתונים של Oracle, חלים כללים מיוחדים לטיפול במקור נתונים ולמיטוב ביצועי השאילתה. ארבעת נתיבי הגישה הבסיסיים לנתונים מרוחקים, כוללים טבלאות צמודות, שאילתות מעבר (pass-through queries) SQL, ODBCDirect ו- OLE DB.

בעבודה עם Access 2000, קרוב לוודאי שתעדיף להימנע מגישה לנתונים באמצעות ODBCDirect. עדיף ללמוד להשתמש בספקי ADO ו-OLE DB, מכיון ש- ADO/OLE DB מהווה חלק מגישת הנתונים האוניברסלית של Microsoft (Microsoft Universal Data Access). הטכנולוגיה ODBCDirect הולכת והופכת מיושנת, ו- OLE DB עתיד להחליף אותה.

שאילתות מעבר SQL הן שאילתות המבוססות על תחביר SQL טבעי של מנגנון מסד נתונים מרוחק. סוג שאילתה זה מאפשר לעבוד ישירות עם טבלאות בשרת המרוחק במקום לקשר אותן תחילה. העובדה שניתן להשיג תוצאה זהה בעזרת ADO עם תחביר בעל אפשרות העברה טובה יותר הופכת גם את שאילתות המעבר לאמצעי מיושן.

עבודה עם טבלאות מקושרות יכולה לאפשר ליישום גישה מבוססת-מטמון לרשומות במקור נתונים מרוחק. הדבר עלול להאט את מהירות עבודתו של יישום, אך בו-בזמן גם להאיץ שאילתות שיש להן מקורות נתונים מקושרים, במיוחד כשהנתונים אינם משתנים לעיתים תכופות.

במהלך עיצוב שאילתות למקורות נתונים מרוחקים, חשוב לתכנן אותן כך שתבצענה את רוב השאילתה במחשב המפעיל את מנגנון מסד הנתונים המרוחק (השרת), שביצועיו עולים לאין שיעור על אלה של מחשב שולחני טיפוסי. בנוסף, הפעלת השאילתה בשרת מאפשרת להקטין את תעבורת הרשת ולהעביר רק חלק מהנתונים ברשת. לפניך עצות אחדות למיטוב שאילתות המבוצעות על מקור נתונים מרוחק:

- הגבלת פונקציות צבירה SQL לאלו שמנגנון מסד הנתונים המרוחק תומך בהן (הפונקציות COUNT, SUM, MIN, MAX ו-AVG)
- הגדרת סוג הנעילה adLockBatchOptimistic והפעלת השיטה UpdateBatch כדי למטב את העבודה עם מקור נתונים מרוחק בצורה לא מחוברת
- הימנעות מהפעלת פונקציות מותאמות אישית שדורשות משאבי עיבוד מקומיים
- שימוש בקריטריונים המגדירים טווח קבוע, כגון בין 100 ל-1,000, לעומת קריטריונים "פתוחים", כגון >100
- מילוי חוקים כלליים של מיטוב שאילתות, מיון, חיפוש וצירוף בשדות סדורי אינדקס, ושימוש בפרמטרים של תווים כלליים (wildcard) עם האופרטורים LIKE רק בסוף קריטריון חיפוש מחרוזת



הערה:

SQL Server תומך בתו % במקום בתו * לחיפושים המבוססים על תווים כלליים עם האופרטור LIKE. אם השאילתה מופעלת מול מנגנון מסד נתונים SQL Server הכולל את Microsoft Data Engine החדש, הקפד לשלב בשאילתות את התו הכללי החדש.

עיצוב שאילתות באופן ידני

יישומים מותאמים אישית יכולים לבנות שאילתות באמצעות קוד, או לנצל שאילתות בנויות מראש בתור מקורות של רשומות. בדיוק כמו בטבלאות, סביר מאוד שלעיתים תבנה שאילתות באופן ידני ולעיתים – באמצעות קוד. ידע מבוסס בעקרונות עיצוב ידני של שאילתה יהיה לך לעזר רב בבואך ללמוד עיצוב והפעלה של שאילתות באמצעות קוד.

בסעיף זה נבחן לעומק בניית שאילתות באמצעות אשפים ונסקור את התצוגה **עיצוב** (Design) של שאילתה. קל מאוד להשתמש באשפים, אך הם מגבילים במידה רבה את אפשרויות העיצוב בהשוואה לבניית שאילתות באופן ידני ברשת עיצוב השאילתה שבתצוגת **עיצוב**. סקירת טכניקות השימוש ברשת עיצוב השאילתה כוללת: הוספת טבלאות, בחירת שורות וציון קריטריונים. כאן תגלה גם כיצד להרכיב שאילתות פרמטר, שאילתות איחוד ושאילתות משנה.

שימוש באשפים

אשפי השאילתות של Access מאפשרים ליצור שאילתות בצורה קלה ומהירה. סעיף זה מתאר את ארבעת האשפים.

אשף השאילתה הפשוטה

החלון **מסד נתונים** (Database) של Access 2000 עוצב מחדש באופן שהופך את הגישה **לאשף שאילתה פשוטה** (Simple Queries Wizard) לפעולה ייחודית. בחר את **שאילתות** (Queries) בחלון **מסד נתונים** ולחץ לחיצה כפולה על הסמל **יצירת שאילתה באמצעות אשף** (Create Query By Using Wizard) כדי לפתוח את האשף. באפשרותך גם להגיע אל אשף זה בדרך המקובלת יותר, על ידי בחירה באפשרות **שאילתות** ולחיצה על לחצן **חדש** (New) בסרגל הכלים של החלון **מסד נתונים**.

אשף השאילתה הפשוטה מאפשר שלושה סוגי עיצוב שאילתה. כל רכיביו מיועדים למפתחים מתחילים. יחד עם זאת, גם מפתחים מנוסים ייווכחו שהוא מאפשר ליצור שאילתות במהירות המשתווה לזו של עבודה עם רשת עיצוב השאילתה. חסרונו

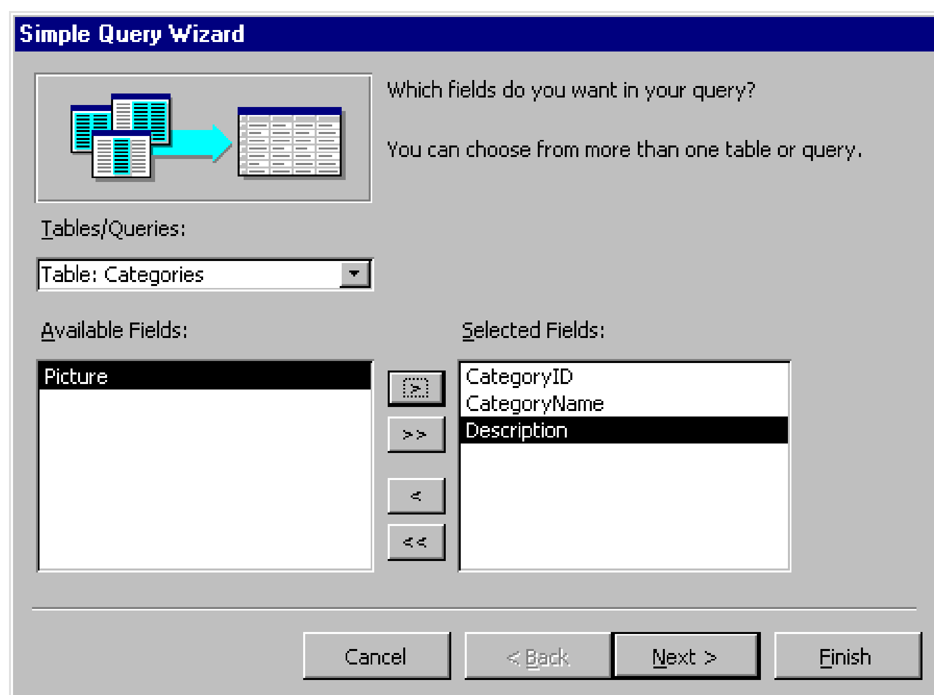
העיקרי של אשף השאילתה הפשוטה הוא שאינו מאפשר את טווח עיצובי השאילתה המלא שניתן להשיג באמצעות רשת עיצוב השאילתה או SQL. מפתחים השולטים ב-SQL ובכללי הצירוף (joining rules), יחבבו את הממשק הגרפי שלו.



הערה:

היישום החדש Microsoft Access Project אינו כולל אשפי שאילתה כלשהם (סוג פרויקט זה יוצר קבצים מסוג *.adp בניגוד לקבצי מסד הנתונים *.mdb המקובלים). בנוסף, עיצוב התצוגה **עיצוב** עבור קבצי *.adp שונה מזה של קבצי *.mdb. עיין בסעיף האחרון בפרק זה לקבלת הסבר ראשוני בעיצוב שאילתה בעזרת Access Projects. פרק 12 מרחיב בנושא העבודה עם מסדי נתונים מרוחקים ו-Microsoft Data Engine.

הדרך הבסיסית ביותר לבנות שאילתה בעזרת אשף השאילתה הפשוטה (ראה תרשים 4.1) היא לבסס אותו על טבלה או על שאילתה יחידה. הפעל את האשף באופן זה כדי להפריד בין השדות שהשאילתה מחזירה. למרות פשטות השימוש באשף, תפיק ממנו תועלת רבה, מכיון שהגבלת קבוצת ההחזרה משפרת את ביצועי השאילתה.

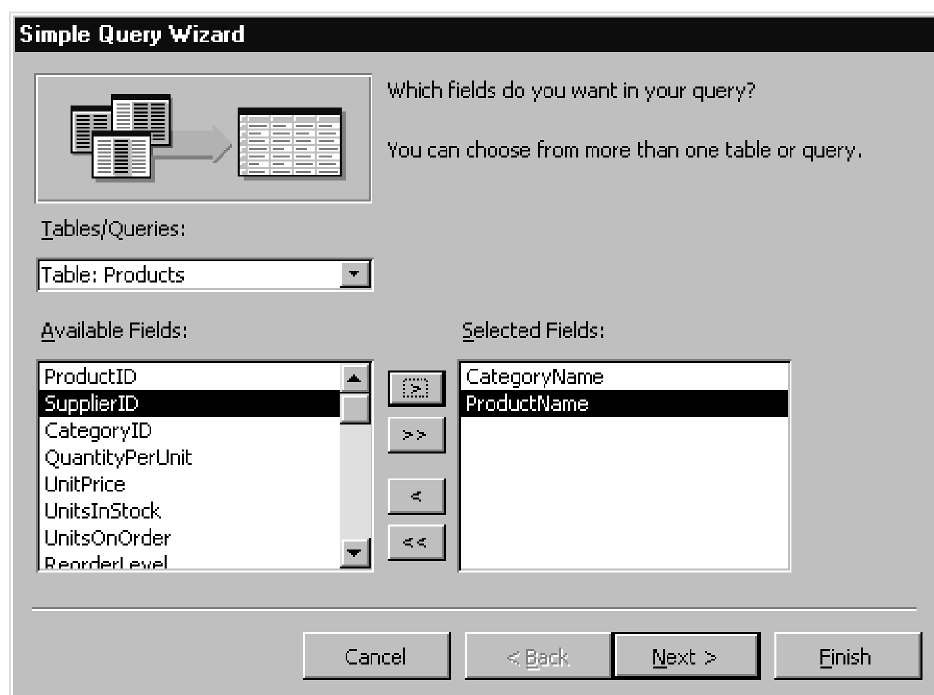


תרשים 4.1: אשף שאילתה פשוטה במהלך יצירת שאילתה המבוססת על קבוצת משנה של שדות בטבלה Categories במסד הנתונים Northwind

עוצמתו האמיתית של אשף השאילתה הפשוטה באה לידי ביטוי ביכולתו לצרף שתי טבלאות או יותר, מבלי לחייב את מעצב השאילתות ליצור צירופים (joins) בפועל בין הטבלאות. הדבר מאפשר למעצב להתמקד בשדות הדרושים בשאילתה ובטבלאות שמהן יילקחו השדות, מבלי להקדיש זמן כלשהו לעיצוב הנאות של שאילתה. האשף נבון במידה הדרושה ליצור תוצאות תקינות.

תרשים 4.2 מציג את אשף השאילתה הפשוטה בשלבי הפיתוח של שאילתה המצרפת שתי טבלאות במסד הנתונים Northwind. כל שעליך לעשות הוא לבחור טבלה ולאחר מכן לבחור שדות בטבלה זו ולהעבירם מתיבת הרשימה **שדות זמינים** (Available Fields) אל התיבה **שדות שנבחרו** (Selected Fields). החלון התחתון מציג קטע מקבוצת התוצאה של צירוף השדה CategoryName מהטבלה Categories והשדה ProductName מהטבלה Products.

השאילתה שבתרשים 4.2 מצרפת את הטבלאות Categories ו-Products בשדה המשותף שלהן, CategoryID, ואולם, בשום שלב אינך צריך להגדיר זאת. אשף השאילתות הפשוטות מזהה בעצמו את השדה המשותף ומצרף באמצעותו את שתי הטבלאות. האשף מניח אוטומטית צירוף פנימי הכולל רק רשומות בעלות ערכים תואמים של השדה CategoryID בשתי הטבלאות. אם רשומה באחת הטבלאות מכילה ערך CategoryID שלא קיים ברשומה בטבלה המקבילה, אשף השאילתה הפשוטה יתעלם ממנה. אם היישום שלך אמור להכיל רשומות שקיימות רק בטבלה אחת ללא הקבלה בטבלה השנייה, תוכל לערוך את עיצוב השאילתה מתוך האשף, לעצב את השאילתה בתצוגת **עיצוב** או להרכיב אותה בעזרת משפטי SQL.



Categories Products List Query : Select Query

Category Name	Product Name
Beverages	Chai
Beverages	Chang
Beverages	Guaraná Fantástica
Beverages	Sasquatch Ale
Beverages	Steeleye Stout
Beverages	Côte de Blaye
Beverages	Chartreuse verte
Beverages	Ipoh Coffee
Beverages	Laughing Lumberjack Lager
Beverages	Outback Lager
Beverages	Rhönbräu Klosterbier
Beverages	Lakkalikööri
Condiments	Aniseed Syrup
Condiments	Chef Anton's Cajun Seasoning

Record: 1 of 77

תרשים 4.2: תיבת הדו-שיח של אשף השאילתה הפשוטה ותצוגת גיליון הנתונים המתקבלת ממנה

באפשרותך גם לעצב שאילתות המבצעות פעולות צבירה מסוימות בעזרת אשף השאילתה הפשוטה. האשף מזהה בעצמו אם ניתן לבצע צבירה ועושה ככל האפשר לסייע למעצב השאילתה. האשף מציג תיבת דו-שיח כמו זו שבחלקו העליון של תרשים 4.3 רק כשניתן ליצור צבירות מספריות. ניתן עדיין לצמצם את נתיב הצבירה על ידי בחירה באפשרות **פירוט** (Detail) בשלב השני של האשף. אם תבחר באפשרות **סיכום** (Summary), יהיה עליך גם ללחוץ על לחצן **אפשרויות סיכום** (Summary Options) כדי לציין את השדות שייצברו ואת הפונקציות שבאמצעותן תתבצע הצבירה. האשף מכין לפעמים שדה סיכום נוסף. אם הוא עושה זאת, מחק את השדה ברשת עיצוב השאילתה.

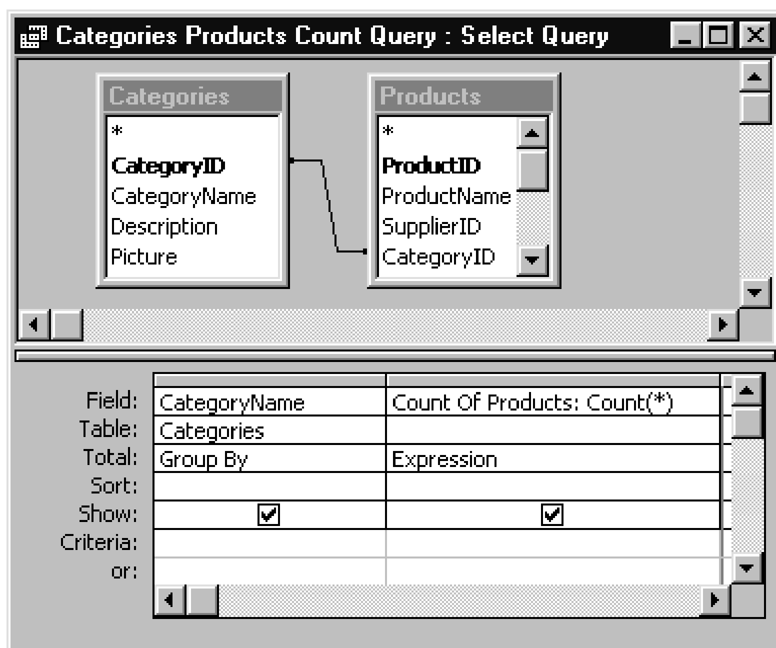
Simple Query Wizard

Would you like a detail or summary query?

- Detail (shows every field of every record)
- Summary

Summary Options ...

Cancel | < Back | Next > | Finish



Categories Products Count Quer...

Category Name	Count Of Products
Beverages	12
Condiments	12
Confections	13
Dairy Products	10
Grains/Cereals	7
Meat/Poultry	6
Produce	5
Seafood	12

Record: 1 of 8

תרשים 4.3: שלב 2 של תיבת הדו-שיח **אשף שאילתה פשוטה** מאפשר לבצע חישובי צבירה בשאילתה

שני החלונות התחתונים של תרשים 4.3 מציגים את תצוגות העיצוב וגיליון הנתונים של השאילתה. שים לב שהשאילתה מונה את המוצרים בכל קטגוריה. החלון האמצעי מציג את רשת עיצוב השאילתה שמחשבת את התוצאה. לעומת זאת, אשף השאילתה הפשוטה מבטל את הצורך לדעת היכן להציב את פונקציית הצבירה ואת הבחירות שיש לבצע בשורה **סך הכל** (Total) (עליך ללחוץ על הלחצן **סכומים** (Totals) בסרגל הכלים כדי להציג את השורה **סך הכל** ברשת עיצוב השאילתה בעת עיצוב ידני של שאילתה). האשף משחרר אותך מכל הפעולות האלו.

אשף שאילתות חיפוש כפילויות

צמד אשפי שאילתות מסייע ליצור שאילתות המבצעות משימות שכיחות. **אשף שאילתת חיפוש כפילויות** (Find Duplicates Query) מחפש בערכת רשומות כדי לקבוע אם קיימות שתי רשומות או יותר המכילות ערכים זהים. האשף מחזיר את הערכים הכפולים של כל הרשומות בעלות ערכים משותפים בשדות שצוינו בקריטריוני החיפוש.

אחת הסיבות השכיחות לחיפוש כפילויות היא ביטול עותקי רשומות נוספים במקור רשומה. יחד עם זאת, לא ניתן להפוך את השאילתה שיצר האשף לשאילתת מחיקה, מכיון שהיא תסיר את הרשומות המקוריות יחד עם בנות זוגן. מערכת העזרה המקוונת של Access 2000 מתארת שיגרה להסרה אוטומטית של כפילויות בהתאם לתוצאה שהתקבלה מאשף שאילתת חיפוש כפילויות. כדי להציג את השיגרה, הקלד "מחק רשומות כפולות" (automatically delete duplicate) בתור קריטריון החיפוש בכרטיסיה **חיפוש** של העזרה (שם המאמר: **מחק באופן אוטומטי רשומות כפולות מטבלה**).

פתח את ההוראות הספציפיות על ידי בחירה בנושא **מחק באופן אוטומטי רשומות כפולות מטבלה**.

הדוגמה שלנו לאשף זה מבוססת על הטבלה FamilyMembers (עיין בחלון העליון בתרשים 4.4). תיבת הדו-שיח **אשף שאילתת חיפוש כפילויות** במרכז התרשים, מכילה זוג תיבות רשימה לבחירת שדות בהם יש לחפש ערכים כפולים. כמו במקרה של אשף השאילתה הפשוטה, מעבירים שדות מתיבת הרשימה **שדות זמינים** אל התיבה **שדות בעלי ערכים כפולים** (Duplicate-Value Fields). בדוגמה, נבחר השדה Lname שבאמצעותו יתבצע חיפוש אחר כפילויות. כפי שנראה בחלון העליון בתרשים 4.4, כל הרשומות שבטבלה FamilyMembers הן כפילויות של ערך אחד או יותר של שדה זה. שלב 3 של האשף מאפשר לייעד שדות נוספים שיוצגו בקבוצת ההחזרה. הדוגמה בוחרת בכל השדות הנותרים.

החלון התחתון בתרשים 4.4 מציג את אחד הצדדים החזקים ביותר של אשפי Access: פישוט משימות מורכבות. החלון התחתון מציג גם את עיצוב רשת עיצוב השאילתה שיוצר האשף. שים לב שהתצוגה כוללת שאילתת משנה ב-SQL. משפט שאילתת המשנה כולל פונקציית צבירה וגם את הפסוקיות GROUP BY ו-HAVING. זו שאילתת המשנה בשורת הקריטריון של רשת העיצוב שמזהה ערכים כפולים בערכת הרשומות של המקור. באפשרותך לכוונן את עיצוב השאילתה שנוצרת על ידי הוספה או הסרה של שדות, כולל שדות מקור חדשים או חליפיים, או אפילו על ידי שימוש בקריטריונים נוספים להסרת כפילויות מסוימות.

FamilyMembers : Table

FamID	Fname	Lname	Relation
1	Rick	Dobson	Me
2	Virginia	Dobson	wife
3	Glen	Hill	son
4	Tony	Hill	son
5	Shelly	Hill	daughter-in-law

Record: 1 of 5

Find Duplicates Query Wizard

Which fields might contain duplicate information?

For example, if you are looking for cities with more than one customer, you would choose City and Region fields here.

Available fields: FamID, Fname, Relation

Duplicate-value fields: Lname

Cancel | < Back | Next > | Finish

Find duplicates for FamilyMembers : Select Query

FamilyMembers: *, FamID, Fname, Lname, Relation

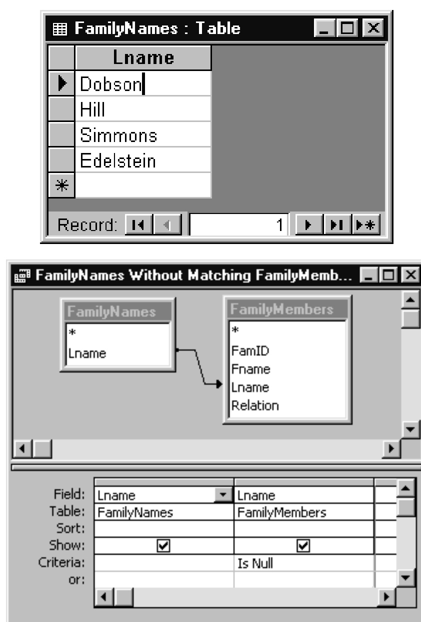
Field:	Lname	FamID	Fname
Table:	FamilyMembers	FamilyMembers	FamilyMembers
Sort:	Ascending		
Show:	☑	☑	☑
Criteria:	In (SELECT Lname FROM FamilyMembers As Tmp GROUP BY Lname HAVING Count(*)>1)		
or:			

תרשים 4.4: אשף שאילתת חיפוש כפילויות. החלון העליון מציג את מקור רשומות הקלט. תיבת הדו-שיח המרכזית מציגה בחירת שדה שלפיו יתבצע חיפוש הכפילויות. החלון התחתון מציג את השאילתה שנוצרה בתצוגת **עיצוב**

אשף שאילתת חיפוש של לא תואמים

אשף שאילתת חיפוש של לא תואמים (Find Unmatched Query Wizard) הוא רכיב נוסף התומך במטלת ניהול מקובלת של מסדי נתונים. האשף מחזיר רשומות בטבלה אחת שאין להן רשומות תואמות בטבלה מקבילה. נוח במיוחד לנהל באמצעותו טבלאות שאין להן שלמות קשרים, אך קיים ביניהם קשר גומלין מסוג יחיד לרבים. במצב זה, באפשרותך להסיר, בשוגג או במתכוון, את צד היחיד או את צד הרבים של קשר הגומלין. כאשר לאחד מצידי הקשר, היחיד או הרבים, אין בן-זוג בטבלה המקבילה, אשף שאילתת חיפוש של לא תואמים יגלה זאת. באופן טיפוסי תרצה להסיר רשומות אלו ממקור הרשומות. אין כל בעיה – כל שעליך לעשות הוא להפוך את השאילתה שנוצרה לשאילתת מחיקה.

השאילתה שבתרשים 4.5 מחפשת רשומות לא תואמות בטבלאות FamilyNames ו-FamilyMembers. ההשוואה מתבצעת על השדה Lname בשתי הטבלאות. השאילתה מצרפת את שתי הטבלאות על ידי אילוץ כל הרשומות מהטבלה FamilyNames לתוך השאילתה, כמצוין בחץ ההכוונה מהטבלה FamilyNames אל הטבלה FamilyMembers. לאחר מכן, השאילתה מחפשת ערכי null בשדה Lname בטבלה FamilyMembers. אם ערך הרשומה של Lname בצד FamilyMembers של קבוצת ההחזרה הוא null, לצד FamilyNames של קבוצת ההחזרה אין בן זוג תואם בטבלה FamilyMembers. העיצוב הנאות של שאילתה זו מחייב הבנה של הקשרים ושל ערכי null. התעמקות בעיצוב שאילתות כגון זו ובשאילתות שמחזירות ערכים כפולים עשויה לחדד את מיומנויותיך בעיצוב שאילתות, במקרה שאינך רגיל לכתוב שאילתות כאלו דרך שיגרה.



תרשים 4.5: אשף שאילתת חיפוש של לא תואמים. מקור הרשומות השני הוא הטבלה FamilyMembers. קריטריון ההתאמה הוא השדה Lname.

אשף שאילתה מוצלבת

מנתחי תמיכה בהחלטות מגלים לפעמים ששאילתות הצלבות (crosstab queries) מסייעות להגיע לתובנות חדשות אודות אובייקטים ותהליכים. תרשים 4.6 מציג טבלת קלט של אשף שאילתה מוצלבת ואת קבוצת ההחזרה שהוא יוצר ממנה. תצוגת קבוצת ההחזרה טופלה בתצוגות **עיצוב** (Design) **וגיליון נתונים** (Datasheet).

שאילתת ההצלבות שבגיליון הנתונים התחתון מונה את FamID לפי השדות Relation ו-Lname של גיליון הנתונים העליון. ה"טיפול" שהענקתי לתצוגת העיצוב מכוונן את מיון השורות לפי Relation, מסדר עולה ליורד. בתצוגה **גיליון נתונים**, גררתי את העמודה המסומנת בתור Total of FamID כך שבמקום שתהיה העמודה השנייה, היא מעתה העמודה האחרונה. אחרת, העיצוב היה אוטומטי. לא היה צורך להכניס שמות שדות ומילות מפתח ברשת עיצוב השאילתה, וגם לא היה צורך לכתוב קוד SQL.

FamilyMembers : Table

FamID	Fname	Lname	Relation
1	Rick	Dobson	Me
2	Virginia	Dobson	wife
3	Glen	Hill	son
4	Tony	Hill	son
5	Shelly	Hill	daughter-in-law

Record: 1 of 5

FamilyMembers_Crosstab : Crosstab Query

Relation	Dobson	Hill	Total Of FamID
wife	1		1
son		2	2
Me	1		1
daughter-in-law		1	1

Record: 1 of 4

תרשים 4.6: שאילתה שנוצרה באמצעות אשף שאילתה מוצלבת של Access 2000. גיליון הנתונים העליון מציג את הקלט של קבוצת ההחזרה של השאילתה שמופיעה בגיליון הנתונים התחתון

שתי טכנולוגיות ניתוח חדשות נוטלות את הבכורה משאילתות ההצלבות. הראשונה, טבלת ציר (pivot table) יכולה לחשב תוצאות הצלבה פשוטות מתוך מקור רשומה, אך באפשרותה גם להפעיל טיפול דינמי בנתונים לאחר חישוב ההצלבה. הדבר מאפשר למנתחי החלטות לתקשר באופן דינמי עם תוצאות הניתוח שלהם. Access הופך טבלת ציר לזמינה באמצעות הטפסים ודפי הגישה לנתונים שלו (Data Access Pages), אך Microsoft Excel הוא הכלי הראשי של Office לביצוע ניתוח טבלת ציר. הטכנולוגיה

השנייה, מסד נתונים עיבוד ניתוחי מקוון – OLAP (Online Analytical Processing) מאפשרת ניתוח נתונים דינמי מול מקורות נתונים גדולים מאוד. טכנולוגיה זו מחייבת שימוש בשרת מסד נתונים מרוחק (אחד או יותר), כגון SQL Server 7 וצרכן OLAP עבור הנתונים. בקרב הרכיבים המקובלים של Office 2000, Excel 2000 הוא המשאב המיועד לתפקיד זה.

שימוש בתצוגה עיצוב

אשפי השאילתות מאפשרים לעצב שאילתות בלי תלות ברמת השליטה בתחביר או בסוגיות העיצוב, אך הן מצמצמים את היקף האפשרויות שניתן להשיג בעזרת שאילתות. כפי שקורה בעת השימוש באשפים, לעיתים הם יוצרים עיצוב מורכב יותר מזה הדרוש בפועל כדי למלא משימה מוגדרת. הדבר עלול לסבך את תהליך עריכת שינויים קטנים בשאילתה. אם תעצב את כל השאילתה בעצמך, קרוב לוודאי שתיזכר בלוגיקה הנכונה ותצליח לערוך בה שינויים בהמשך ללא קושי מיוחד.

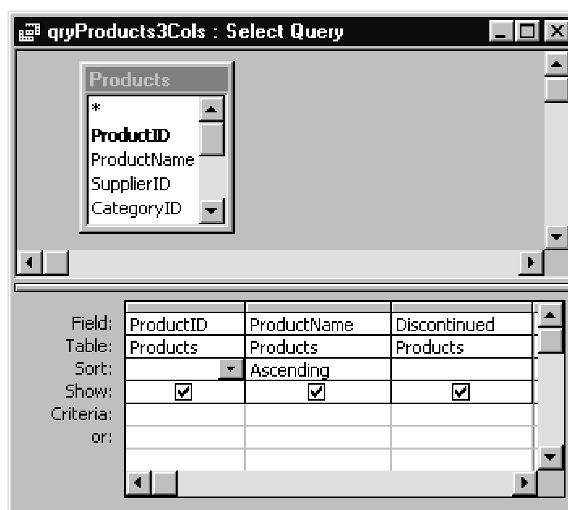
הוספת טבלאות ושאילתות

באפשרותך ליצור בעצמך שאילתות מותאמות אישית על ידי הוספת טבלאות או שאילתות אחרות לרשת עיצוב השאילתה. אם תבחר בדרך זו, בדרך כלל לא תצטרך לכתוב קוד SQL כדי ליצור שאילתות ולערוך אותן. פתח את רשת עיצוב השאילתה על ידי לחיצה כפולה על **יצירת שאילתה בתצוגת עיצוב** (Create Query In Design View) בקבוצת האובייקטים **שאילתות** (Queries) שבחלון **מסד נתונים** (Database). לאחר מכן, הוסף טבלה או שאילתה אחת או יותר, או צירוף כלשהו של טבלאות ושאילתות בראש רשת העיצוב. בחר טבלה או שאילתה ולחץ על **הוסף** (Add) כדי להעביר אותה מתיבת הדו-שיח **הצגת טבלה** (Show Table) לראש רשת העיצוב.

לאחר שהוספת טבלה לראש התצוגה **עיצוב**, עליך לבחור שדות שיהוו חלק מהשאילתה ברשת התחתונה. קיימות לפחות שלוש דרכים להוספת שדות לשאילתה. דרך אחת – גרירה ושחרור של שדה מתיבת הרשימה שמייצגת את הטבלה בחלק העליון אל השורה **שדה** (Field) ברשת התחתונה. אם תגרור ותשחרר שדה בקצה הימני העליון של עמודה שכבר מאוכלסת בנתונים, Access יעביר אוטומטית את העמודות הנותרות לצד שמאל (בממשק עברי). דרך שנייה – באפשרותך ללחוץ לחיצה כפולה על ערך ברשימת השדות. על ידי כך תעתיק את השדה לעמודה הפנויה הראשונה ברשת. דרך שלישית – ניתן לבחור שדה מתוך תיבה נפתחת בשורה **שדה** של כל אחת מהעמודות. אם חלקה העליון של הרשת מכיל יותר ממקור רשומות אחד, הרשימה הנפתחת של השורה **שדה** תמספר שדות שקובצו לפי מקור רשומות.

באפשרותך לפעול בכל אחת משלוש הדרכים בטיפולך בטבלה בודדת כדי לציין עמודות אחדות בטבלה המכילה עמודות רבות. תרשים 4.7 מציג את התצוגות **עיצוב** ו**גיליון נתונים** (Datasheet) של שאילתה המבוססת על הטבלה Products בדוגמה. השאילתה מציינת שלושה שדות מתוך הטבלה.

השורה **sort** בעמודה ProductName בתרשים 4.7 נקבעה בתור **סדר עולה** (Ascending). שורה זו מכילה רשימה נפתחת ובה שלוש אפשרויות: **סדר עולה**, **סדר יורד** (Descending) **ולא מוין** (Not Sorted). אם תבחר באפשרות האחרונה, התא יופיע כשהוא ריק ברשת ונתוני העמודה לא יהיו ממוינים בעת הצגת השאילתה בתצוגת **גיליון נתונים**. בחירה באפשרות **סדר עולה** בעמודה ProductName גורמת לרשומות להופיע בסדר אלפבתי בהתאם ל-ProductName. אם שורת המיון של ProductName ריקה, גיליון הנתונים בחלון התחתון של תרשים 4.7 יופיע ממוין לפי המפתח הראשי של הטבלה, ProductID.



qryProducts3Cols : Select Query

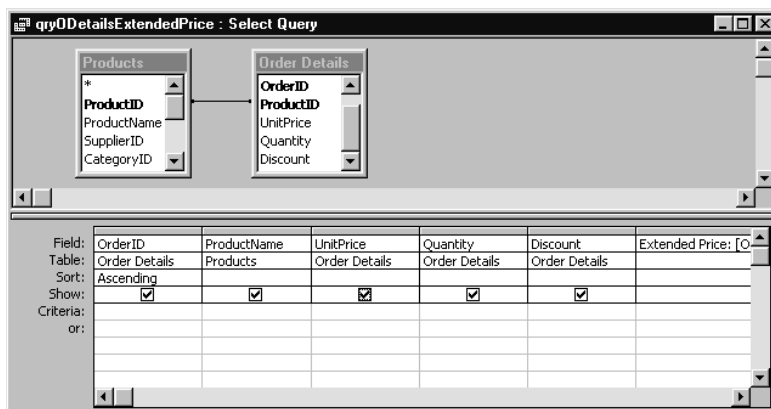
Product ID	Product Name	Discontinued
17	Alice Mutton	☑
3	Aniseed Syrup	☐
40	Boston Crab Meat	☐
60	Camembert Pierrot	☐
18	Carnarvon Tigers	☐
1	Chai	☐
2	Chang	☐
39	Chartreuse verte	☐
4	Chef Anton's Cajun Seasoning	☐
5	Chef Anton's Gumbo Mix	☑
48	Chocolade	☐
38	Côte de Blaye	☐
58	Escargots de Bourgogne	☐
52	Filo Mix	☐

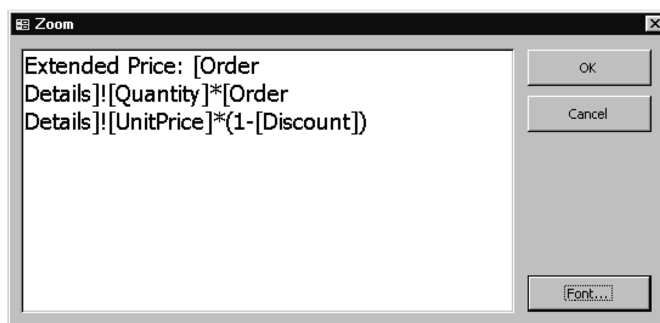
Record: 1 of 77

תרשים 4.7: החלון העליון מציג את תצוגת השאילתה של שאילתה שנוצרה באופן ידני ומבוסס על הטבלה Products. החלון התחתון הוא גיליון הנתונים שהתקבל כתוצאה מהשאילתה

קל ופשוט להוסיף מקור רשומות אחד לתצוגת שאילתה. באפשרותך לבחור מקור רשומות קלט מתוך טבלאות ושאילתות אחרות. מומלץ מאוד לצרף מקורות רשומות (אם Access אינו מצרף אותם בעצמו בהתאם לקשרי הגומלין המצויינים בחלון **קשרי גומלין**). כשלון צירוף הטבלאות יגרום לכך שגיליון הנתונים שיתקבל יציג את התוצאה הקרטזית של שתי הטבלאות (כל השילובים הסדורים האפשריים של שורות מהטבלה הראשונה עם כל השורות מהטבלה השנייה) של שתי הטבלאות. זו תהיה לעיתים קרובות טבלה בעלת מספר שורות גדול במידה מופרזת.

תרשים 4.8 מציג את הטבלאות Products ו-Order Details מצורפות באמצעות השדה ProductID. באפשרותך ליצור צירוף ברירת מחדל כמו זה על ידי גרירת שדה מטבלה אחת ושחרורו בשדה התואם בטבלה השנייה. כך תקבל גיליון נתונים עם שורה אחת לכל שדה תואם בשתי הטבלאות. אם רשומה כלשהי מופיעה בטבלה אחת בלבד, היא לא תיראה בגיליון הנתונים של השאילתה. קיימים שני סוגי צירוף נוספים. סוג אחד מצרף את כל השורות מהטבלה השמאלית Products, והאחר מצרף את כל השורות מהטבלה הימנית Order Details. באפשרותך לבחור אחד מסוגי הצירוף הללו על ידי לחיצה ימנית על קו הצירוף ובחירת סוג הצירוף המבוקש לאחר מכן.





תרשים 4.8: החלון העליון מציג את עיצוב השאילתה של שאילתה שנוצרה באופן ידני בהתאם לשתי הטבלאות, Products ו- Order Details. החלון התחתון מציג את תיבת הדו-שיח **שנה מרחק מתצוגה** (Zoom) המציגה את הנוסחה של שדה מחושב בשאילתה

השורה **טבלה** (Table) בתרשים 4.8 מציגה ארבע עמודות ששייכות לטבלה Order Details. עמודה נוספת, ProductName, שייכת לטבלה Products. השדה האחרון של עיצוב השאילתה אינו נראה בשלמותו בחלון העליון של התרשים.

התייחסות לגליונות נתונים משניים

אחד החידושים שהונהגו ביישום Access 2000 הוא היכולת להתייחס בצורה היררכית לטבלה או לשאילתה אחת מתוך שאילתה אחרת. במובן מסוים אפשר לומר שלשאילתה אם יש טבלה או שאילתה בת.

תרשים 4.9 מציג גיליון נתונים משני בתוך גיליון נתונים של שאילתה. השאילתה הָאֵם נקראת qryCategories. שים לב לסימן החיבור (+) בעמודה הראשונה של השאילתה. לחיצה על אחד הסימנים הללו, למשל הקטגוריה **Meat/Poultry**, פותחת גיליון נתונים בן והופכת את סימן החיבור לסימן חיסור (**-**). הסימן המתחלף מתפקד כמתג דו-מצבי לפתיחה וסגירה של גיליון נתונים משני המקושר לשורה כלשהי בשאילתת האם. גיליון הנתונים המשני (הבן) שבתרשים מציג את שדות המוצר עבור שורה אחת של גיליון הנתונים האב שלו. כללית ניתן לומר שגיליון הנתונים הבן מפנה לשאילתה או לטבלה נוספת שמקושרת לשורה הנוכחית. באפשרותך לפתוח בו-זמנית גליונות נתונים בנים רבים. לאמיתו של דבר, מאפיין של שאילתה מאפשר להשאיר את כל גליונות הנתונים הבנים פתוחים כברירת מחדל.

qryCategories : Select Query

Category ID	Category Name	Description	Picture
1	Beverages	Soft drinks, coffees, teas, beers, and ales	Bitmap Image
2	Condiments	Sweet and savory sauces, relishes, spreads, and seasonings	Bitmap Image
3	Confections	Desserts, candies, and sweet breads	Bitmap Image
4	Dairy Products	Cheeses	Bitmap Image
5	Grains/Cereals	Breads, crackers, pasta, and cereal	Bitmap Image
6	Meat/Poultry	Prepared meats	Bitmap Image

Product ID	Product Name	Supplier	Category	Quantity Per Unit	Unit Price
9	Mishi Kobe Niku	4	Meat/Poultry	18 - 500 g pkgs.	$97.00
17	Alice Mutton	7	Meat/Poultry	20 - 1 kg tins	$39.00
29	Thüringer Rostbratwurst	12	Meat/Poultry	50 bags x 30 sausgs.	$123.79
53	Perth Pasties	24	Meat/Poultry	48 pieces	$32.80
54	Tourtière	25	Meat/Poultry	16 pies	$7.45
55	Pâté chinois	25	Meat/Poultry	24 boxes x 2 pies	$24.00
utoNumber)			Meat/Poultry		$0.00

Record: 1 of 6

Category ID	Category Name	Description	Picture
7	Produce	Dried fruit and bean curd	Bitmap Image
8	Seafood	Seaweed and fish	Bitmap Image
(AutoNumber)			

Record: 1 of 8

תרשים 4.9: שאילתת אם, qryCategories, מציגה את גיליון הנתונים המשני שלה עבור הקטגוריה **Meat/Poultry**

חמישה מאפייני שאילתה חדשים שולטים במראה ובתפקוד של גליונות נתונים משניים. המאפיין **שם גיליון נתונים משני** (Subdatasheet Name) מצביע על טבלה או שאילתה המכילות את נתוני הבן. כדי לגשת למאפיין זה או לארבעת האחרים, פתח את תיבת הדו-שיח **מאפיינים** (Properties) בתצוגת **עיצוב** של השאילתה. חמשת

המאפיינים החדשים מופיעים בתחתית תיבת הדו-שיח. תיבת המאפיין **שם גיליון נתונים משני** מכילה תיבה נפתחת שמפשטת את תהליך הבחירה של שאילתה או טבלה אחרת בתור מקור רשומות עבור גיליון נתונים משני. במקרה של השאילתה שבתרשים 4.9, **שם גיליון נתונים משני** הוא הטבלה MyProducts. הכנס שמות שדות עבור המאפיינים **קישור שדות צאצא** (Link Child Fields) **וקישור שדות מראשי** (Link Master Fields) כדי לסנכרן את שאילתת האם עם שאילתת הבן או טבלת הבן. כמו במקרה של טופס עיקרי עם טופס משני, שמות השדות אינם חייבים להיות זהים, אך סוג הנתונים שלהם צריך להיות זהה. המאפיין **גובה גליון נתונים משני** (Subdatasheet Height) שולט בגובה גיליון הנתונים המשני בתצוגת **גיליון נתונים** של גיליון האם. ואחרון חביב, המאפיין **גליון נתונים משני מורחב** (Subdatasheet Expanded) מקבל את הערכים **כן** (Yes) **ולא** (No). תיבת הדו-שיח **מאפייני שאילתה** (Query Properties) כוללת רשימה נפתחת להכנסת אחד משני ערכים אלה.

Access 2000 תומך בגליונות נתונים צאצאים עבור גליונות נתונים הורים המבוססים על טבלאות כמו גם על שאילתות. לעומת זאת, מנגנון התמיכה בגליונות נתונים משניים עבור שאילתות הורים שונה במקצת עבור טבלאות הורים. בטבלה אם ניתן לקבוע את קשר הגומלין בין טבלה אם והטבלה או השאילתה הבת שלה בחלון **קשרי גומלין** (Relationships). אם תבחר דוקא להשתמש בתיבת הדו-שיח **מאפייני טבלה** (Table Properties), עליך לפתוח את הטבלה בתצוגת **עיצוב**. לאחר מכן, עליך לפתוח במפורש את תיבת הדו-שיח **מאפייני טבלה** על ידי לחיצה ימנית בחלקה העליון של הרשת ובחירה ב**מאפיינים** (Properties). פעולה זו מאפשרת להשתמש באותם חמשת המאפיינים כמו עבור שאילתה כדי לקבוע את קשרי הגומלין בין טבלה אם לגיליון הנתונים המשני שלה.

שימוש בשדות מחושבים

באפשרותך ליצור שדה מחושב הגוזר את ערכו מתוך שדה אחד או יותר של שאילתה. חשב שדות אלו באמצעות אופרטורים חשבוניים או אופרטורי מחרוזת, פונקציות מוכללות או פונקציות מותאמות אישית.

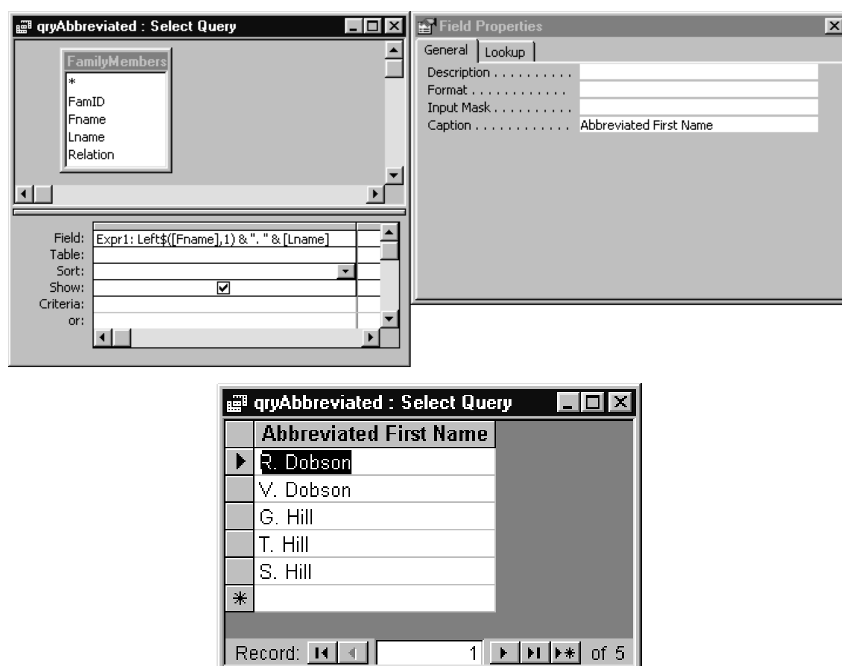
שדות מחושבים אינם ניתנים לעדכון באופן ישיר, מכיון ש-Access מאחסן רק את הביטוי של משפט SQL ולא את תוצאת החישוב. באפשרותך לשנות שדות מחושבים על ידי שינוי הקלטים לביטוי החישוב. למרות ששדות מחושבים אינם ניתנים לעדכון, היישום יכול לצבור אותם לאורך הרשומות.

הגדרת שדות בתור ביטויים בשאילתה טומנת כמה יתרונות. ראשית, מאחר שביטוי יכול להציג נתונים מבלי לאחסן אותם בפועל, שימוש בביטוי עבור שדה מאפשר לשאילתה להציג תוצאה ללא צריכת שטח אחסון. שנית, ביטויים מחושבים מחדש באופן אוטומטי. כך הם יכולים לשקף את הנתונים העדכניים ביותר ללא התערבות כלשהי של מנהל מסד הנתונים. שלישית, קיים מיגוון אפשרויות להכללת ביטויים בתור שדות. לימוד האפשרויות הללו יגדיל את הבנתך הכוללת בעבודה עם Access.

אופרטורים חשבוניים תרשים 4.8 מציג שדה מחושב שמחשב מחיר סיכום לשורה. ביטוי מחיר הסיכום לשורה מתבסס על אופרטור חשבוני פשוט, כגון כפל וחילוק. שים לב שהשורה **טבלה** (Table) של השדה המחושב בחלון העליון של תרשים 4.8 ריקה. משמעות הדבר היא שהשדה אינו תופס שטח אחסון כלשהו. השדה המחושב שבתרשים 4.8 משתמש בתווית שדה לייצוג תווית בעל משמעות של השדה בתצוגת **גיליון נתונים**.

פונקציות מוכללות תרשים 4.10 מציג שדה מחושב שמתבסס על פונקציות מוכללות. הביטוי מרכיב מחרוזת, ולכן באפשרותו להשתמש בפונקציה Left$. השימוש בפונקציה זו חוסך זיכרון בהשוואה לפונקציה הכללית Left שמחזירה מחרוזת שסוג הנתונים שלה משתנה.

הביטוי בתרשים 4.10 משרשר שלוש מחרוזות נפרדות. ראשית, הוא מאחזר את האות הראשונה בשמו הפרטי של בן המשפחה באמצעות הפונקציה Left$. שנית, הוא מצרף את המחרוזת ". " כדי להוסיף נקודה ורווח לאחר האות הראשונה של שם האיש. הביטוי מסתיים בצירוף שם המשפחה. לשדה מחושב זה יש עדיין תווית ברירת מחדל, אך תצוגת **גיליון נתונים** מציגה את העמודה עם כותרת שונה, תיאורית יותר. תיבת הדו-שיח הימנית העליונה בתרשים 4.10 מלמדת שניתן להשיג תוצאה זו על ידי מתן ערך לכיתוב של השדה. ערך כלשהו בתיבת מאפיין הכיתוב יחליף בתצוגה את שם ברירת המחדל של העמודה.



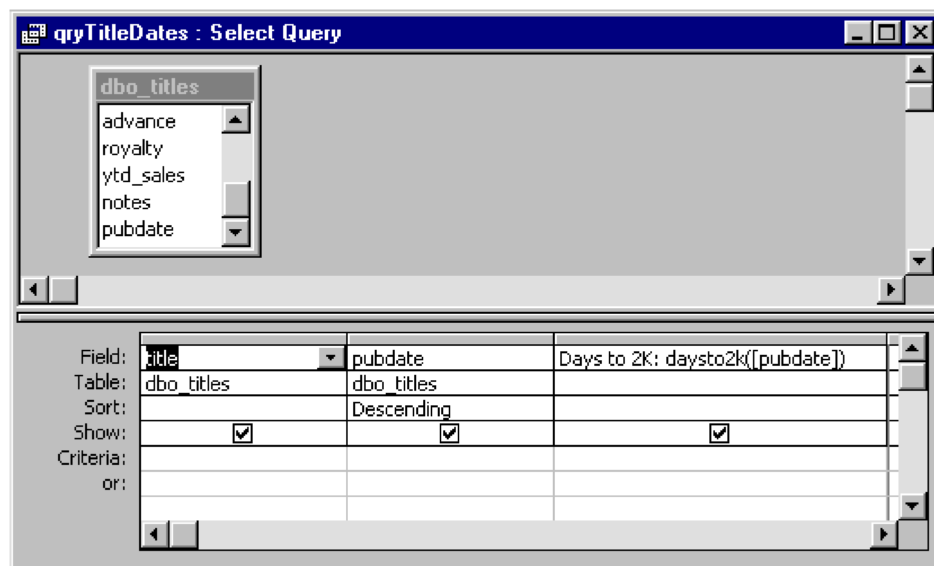
תרשים 4.10: שלושת החלונות שבתרשים חושפים את כללי התחביר והתווית החלים על הוספת ביטוי מחרוזת לשאילתה, במטרה להגדיר את השדה היחיד שלה. החלון התחתון מציג את קבוצת ההחזרה מתוך השאילתה.

ניתן לפשט את תהליך השימוש בפונקציות מוכללות ולהפנות לערכי טבלה, שאילתה וטופס, באמצעות תיבת הדו-שיח **בונה ביטויים** (Expression Builder). אם תמצא לנחוץ לשפר את הביטוי לאחר הוספתו באמצעות בונה הביטויים או באופן ידני, השתמש בתיבת הדו-שיח **שנה מרחק מתצוגה** (Zoom). באפשרותך לעשות זאת על ידי לחיצה ימנית בשורה **שדה** (Field) ובחירה באפשרות **מרחק מתצוגה**. תיפתח תיבת הדו-שיח **שנה מרחק מתצוגה** ובה הביטוי שכתבת. שנה את הביטוי לפי הצורך בטרם תלחץ על **אישור** כדי לסגור את תיבת הדו-שיח ולהכניס את הביטוי שערכת לשאילתה. החלון התחתון שבתרשים 4.8 הוא תיבת שינוי המרחק מתצוגה שמציגה שדה מחושב מתוך שאילתה.

פונקציות מותאמות אישית נוסף לשימוש באופרטורים ובפונקציות מוכללות, באפשרותך לפתח בעצמך פונקציות מותאמות אישית שיחזירו ערכי שדה. תרשים 4.11 מציג שאילתה המקושרת לטבלה dbo_titles במסד הנתונים Pubs של SQL Server.

השאילתה מציגה את השדות title ו-pubdate של כל אחת מהרשומות. בנוסף, היא כוללת שדה מחושב שקורא לפונקציה המותאמת אישית daysto2K. הביטוי מעביר את ערך השדה pubdate אל הפונקציה. הפונקציה daysto2K מופיעה מעל התרשים. אם תשווה את שם הפונקציה שלפניך עם עיצוב השאילתה שבתרשים 4.11, תיווכח כי הפניות לפונקציה בשאילתה אינן תלויות ברישיות (case sensitive).

```
Public Function DaysTo2K(InDate As Date) As Integer
   DaysTo2K = DateDiff("d", InDate, #1/1/2000#)
End Function
```

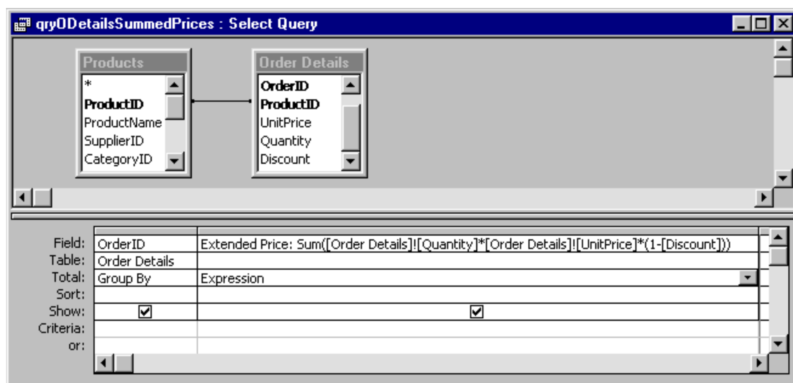


תרשים 4.11: השאילתה שלפניך משתמשת בפונקציה מותאמת אישית לחישוב ההפרש בין ערך שדה בשאילתה לבין התאריך 1 בינואר, 2000

הפונקציה קולטת את ערך השדה pubdate בתור ארגומנט ומחזירה ערך מסוג Integer אל השאילתה. פעולה זו מאפשרת יישור אוטומטי לימין (אם לא מציינים את סוג הנתון המוחזר בתור Integer, הפונקציה תחזיר משתנה (variant) לשאילתה שיגרום ליישור לשמאל). הפונקציה המותאמת אישית מכילה שורה בודדת. היא מיישמת את הפונקציה המוכללת DateDiff כדי לחשב את ההפרש בימים בין השדה pubdate לבין התאריך 1 בינואר, 2000.

יש בעיה ביישום פונקציות מותאמות אישית בביטויים של שדות שאילתה, בעיקר עם מסדי נתונים גדולים ומרוחקים. על היישום להשתהות בזמן שטבלה גדולה עוברת בקווי התקשורת מהשרת המרוחק אל תחנת עבודה מקומית. בנוסף, לא ניתן לנצל את עוצמת החישוב של השאילתה במחשב השרת המרוחק. במצבים בהם הכרחי להפעיל פונקציה מותאמת אישית, הרכב שאילתה שמחלצת את מספר הרשומות המינימלי מהשרת המרוחק ומבצעת את כל שאר הפונקציות בשרת זה. לאחר מכן, החל את הפונקציה המותאמת אישית על קבוצת ההחזרה המצומצמת שהתקבלה מהשרת. הסתייגויות אלו מפונקציות מותאמות אישית אינן נוגעות לדוגמה שבתרשים 4.11, מכיון שהטבלה dbo_titles קטנה מאוד. פונקציות מותאמות אישית פועלות היטב עם מסדי נתונים מבוססי-Jet בכל גודל.

פונקציות צבירה SQL באפשרותך לחשב ערך ללא שימוש בביטוי כלשהו. אחת הדרכים לעשות זאת היא על ידי לחיצה על לחצן **סכומים** (Totals) – הלחצן הנושא את האות סיגמה (∑) – בסרגל הכלים **עיצוב שאילתה** (Query Design). לאחר מכן גרור שדה אחד או יותר אל רשת עיצוב השאילתה ובחר באפשרות **קיבוץ לפי** (Group By) בשורה **סך הכל** (Total). חייב להיות לפחות שדה אחד בעל ערך מספרי שהשורה **סך הכל** שלו אינה מוגדרת בתור **קיבוץ לפי**. באפשרותך לייעד שדה מחושב בתור שדה הערך המספרי. בשורה **סך הכל** בחר ב-Expression. הכנס פונקציית צבירה SQL לשדה המחושב כדי למנות, לסכם או לחשב ממוצע בשדה המחושב עבור כל שילוב ייחודי של שדות שהשורה **סך הכל** שלהן מוגדרת בתור **קיבוץ לפי** במקור הרשומות.



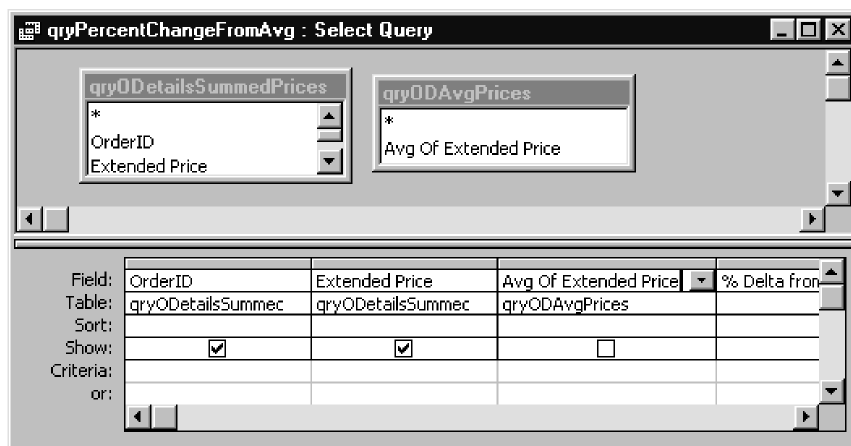
תרשים 4.12: הביטוי בשדה המחושב בשאילתה זו מחשב את מחיר הסיכום לכל שורת פריט בכל הזמנה. מילת המפתח **קיבוץ לפי** בשורה **סך הכל** מאלצת את פונקציית הצבירה שבביטוי לחשב את סכום הערכים על פני שורות הפריטים בכל הזמנה בנפרד

תרשים 4.12 מציג תצוגת עיצוב שמסכמת את מחיר הסיכום (סה"כ) לפי השדה OrderID. השאילתה מחשבת את מחיר הסיכום לכל שורת פריט לפני סיכום וקיבוץ של השורות לפי OrderID. מילת המפתח **קיבוץ לפי** מופיעה בשורה **סך הכל** של העמודה OrderID. הפונקציה Sum בביטוי Extended Price מציינת שהשאילתה תסכם את מחיר הסיכום עבור כל הזמנה.

חישוב ערך מתוך כמה שאילתות יש עניין רב בתוצאות סטטיסטיות שמשוות שני פריטי מידע או יותר. חשוב לדעת את שוויה הכספי של הזמנה, אך לעיתים מעוניינים גם להשוות מחיר ההזמנה למחיר ממוצע ולדעת באיזו מידה הוא גבוה או נמוך ממנו. Access אינו מאפשר לחשב תוצאה כזו בשאילתה בודדת. הדוגמה שנביא בהקשר זה דורשת שלוש שאילתות, כשאחת מהן מכילה תוצאה קרטזית של שתי השאילתות האחרות. השאילתה הראשונה היא זו המוצגת בתרשים 4.12. השאילתה השנייה מחשבת את המחיר הממוצע של כל ההזמנות. שאילתה זו כוללת פונקציית צבירה פשוטה מסוג AVG הפועלת על מחיר הסיכום עבור השאילתה שבתרשים 4.12. החלון העליון בתרשים 4.13 מדגים תוצאה קרטזית של השאילתה שמסכמת את מחירי הסיכום לפי הזמנה (qryODetailsSummedPrice) ושל השאילתה שמחשבת את מחיר הסיכום הממוצע בכל ההזמנות (qryODAvgPrices). כזכור, הדרך לחשב תוצאה קרטזית היא לכלול את שתי השאילתות בתצוגת העיצוב מבלי לצרף אותן. החלון התחתון בתרשים 4.13 מציג את השינוי באחוזים בין הסה"כ של כל הזמנה לבין הממוצע, בכל ההזמנות. אם הסה"כ גדול מהממוצע, ערך האחוז חיובי, אחרת – ערכו שלילי.

הנוסחה שמחשבת את שינוי האחוז מהממוצע אינה מוצגת בתרשים 4.13. הביטוי הוא לא יותר מיחס בין שני מספרים. המונה הוא ההפרש בין מחיר הסיכום בהזמנה לבין המחיר הממוצע, של כל ההזמנות. המכנה הוא מחיר הסיכום הממוצע של כל ההזמנות. מאפיין תבנית העיצוב של שדה מחושב זה מוגדר בתור **אחוז** (Percent) עם 0 מקומות אחרי הנקודה העשרונית.

```
% Delta from Avg: ([Extended Price]-[Avg of Extended Price])/
   [Avg of Extended Price]
```



Order ID	Extended Price	% Delta from Avg
10248	$440.00	-71%
10249	$1,863.40	22%
10250	$1,552.60	2%
10251	$654.06	-57%
10252	$3,597.90	136%
10253	$1,444.80	-5%
10254	$556.62	-64%
10255	$2,490.50	63%
10256	$517.80	-66%
10257	$1,119.90	-27%
10258	$1,614.88	6%
10259	$100.80	-93%
10260	$1,504.65	-1%

qryPercentChangeFromAvg : Select Query — Record: 1 of 830

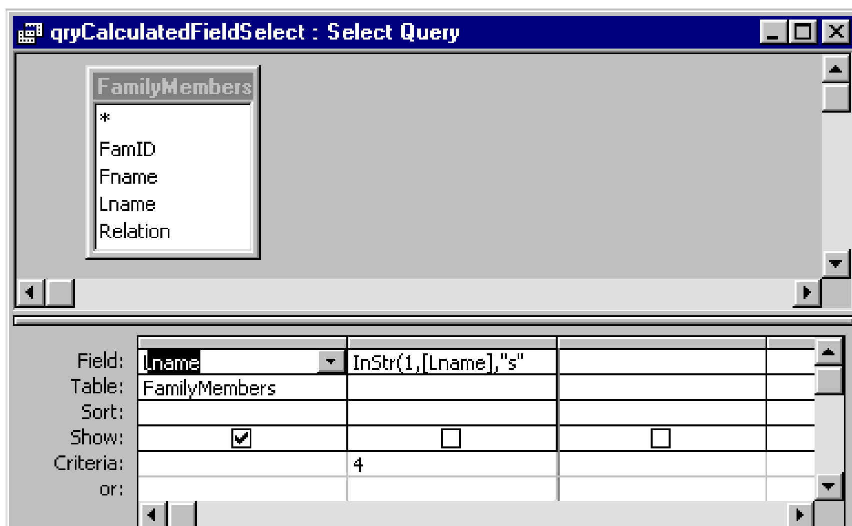
תרשים 4.13: החלון הראשון מציג שאילתה שמחשבת תוצאה קרטזית של שתי שאילתות. הביטוי שבשדה המחושב מחשב את השינוי באחוזים בין מחיר הסיכום להזמנה לבין מחיר הסיכום הממוצע של כל ההזמנות

עבודה עם קריטריונים

התנהגות שאילתות מושפעת בצורה מכרעת על ידי קריטריונים. ציון קריטריונים הכוללים ביטויים וקבועים מאפשר לקבוע את סוג הרשומות שהשאילתה תחזיר. קריטריונים גורמים להכללה או השמטה של רשומות בקבוצת החזרה של שאילתת בחירה. הדוגמאות המוצגות בסעיף זה מתארות שאילתות פשוטות הכוללות תנאי אחד. ברגע שתלמד כיצד ליצור אותן, תוכל לנצל יכולת זו באופן מיידי ליצירת תנאים מורכבים המשלבים שני קריטריונים ואף יותר באמצעות האופרטורים הלוגיים And ו-Or.

שימוש בקריטריון בודד תרשים 4.5 מדגים קריטריון פשוט העושה שימוש במשפט Is Null. משפט זה משמש לזיהוי שדה ריק בשאילתה. עיצוב השאילתה מצרף את הטבלאות FamilyNames ו-FamilyMembers בשדה Lname. סוג הצירוף כולל את כל השמות מתוך הטבלה FamilyNames בין אם הם תואמים לערך Lname בטבלה FamilyMembers ובין אם לאו. בטבלה FamilyNames, השדה Lname מקבל ערך אחד מתוך ארבעה ערכים אפשריים: Dobson, Hill, Simmons ו-Edelstein. השדה Lname בטבלה FamilyMembers מכיל את הערכים Dobson ו-Hill. השאילתה מחזירה את השמות Simmons ו-Edelstein, כיון ששני השמות נמצאים בטבלה FamilyNames אך לא בטבלה FamilyMembers. שינוי המשפט Is Null ל- Is Not Null גורם לקבוצת ההחזרה להשתנות ל- Dobson ו-Hill.

תרשים 4.14 מציג עיצוב שאילתה הכוללת רק את רשומות הטבלה FamilyMembers שהשדה Lname שלהן מכיל את האות הקטנה s במקום הרביעי במחרוזת השם. הפונקציה InStr היא אחת הפונקציות החזקות של Access. הפונקציה משווה בין שתי מחרוזות באחת מכמה דרכים. הפרמטר הרביעי מגדיר את סוג החיפוש; 0 מציין חיפוש בינארי. הפונקציה InStr מחזירה מספר המציין את המיקום של המופע הראשון של המחרוזת השנייה בתוך המחרוזת הראשונה. בביטוי שבתרשים 4.14, InStr מחפשת את המחרוזת s במקום 4 בשם Dobson. מכאן שקביעת הקריטריון 4 תגרום לבחירת רשומות שהשדה Lname שלהן מכיל מחרוזת שבמקום הרביעי שלה נמצאת האות s.

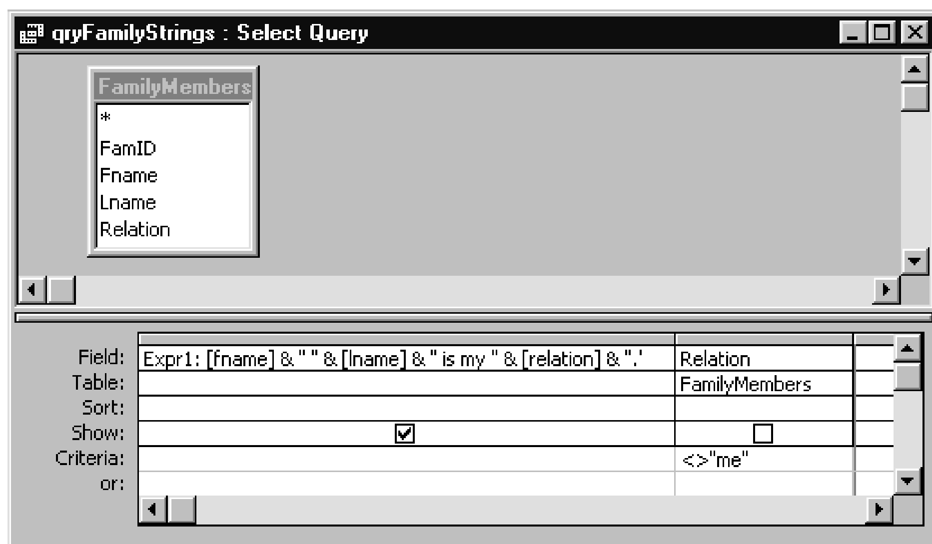


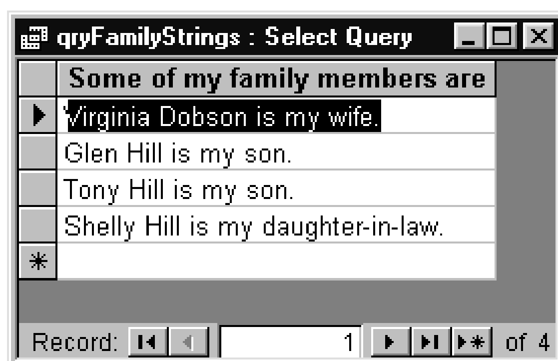
תרשים 4.14: שאילתה זו מחזירה רשומות מהטבלה FamilyMembers שהשדה Lname שלהן מכיל s במקום הרביעי במחרוזת

אם תוסיף לטבלה FamilyMembers רשומות חדשות שהשדה Lname שלהן מכיל מחרוזות שבמקום הרביעי שלה נמצאת האות s, השאילתה תחזיר גם אותן. לדוגמה, הוספת רשומה חדשה שהשדה Lname שלה מכיל את השם Samson, תגרום לשאילתה להחזיר את Samson יחד עם Dobson.

שים לב שתוצאת הביטוי שבתרשים 4.14 אינה מוצגת בקבוצת ההחזרה של השאילתה, מכיון שהתיבה **הצג** (Show) אינה מסומנת.

שימוש בקריטריונים רבים השאילתה שבתרשים 4.15 יוצרת סדרת משפטים. השאילתה מרכיבה את המשפטים באופן דינמי מערכי השדות, אך באותו זמן גם משמיטה את הרשומה שערך השדה Relation שלה הוא me (משפטי קבוצת ההחזרה של השאילתה מתארים את היחס בין כל בן משפחה אלַי (המחבר), ולכן הרשומה שערך השדה Relation שלה הוא me (אני) לא אמורה להיות כלולה בשאילתה. השאילתה משתמשת בשני ביטויים: האחד מציין קריטריון להכללת רשומות, והשני הוא ביטוי מחרוזת שמרכיב את המשפטים באמצעות ערכי שדה נבחרים.





תרשים 4.15: שאילתה שמרכיבה משפטים בצורה דינמית

שימוש בקריטריון מורכב לאחר יצירה ובדיקה של קריטריונים לשדות מסוימים, ייתכן שתמצא לנחוץ ליצור קריטריונים מורכבים שמשלבים שני קריטריונים ויותר לגבי שדות מסוימים. ניתן לעשות זאת בשתי גישות בסיסיות. האחת, לשלב קריטריונים באמצעות האופרטור And (וגם). טכניקה זו גורמת לשאילתה להחזיר שורות של רשומות העומדות בו-זמנית בכל הקריטריונים המצוינים. הגישה השנייה היא לשלב קריטריונים באמצעות האופרטור Or (או). הדבר גורם לשאילתה להחזיר שורות של רשומות העומדות לפחות באחד הקריטריונים מאלה שהוגדרו.

תשמח בוודאי לדעת כי רשת עיצוב השאילתה של Access פוטרת אותך מעיסוק בפרטי התחביר של משפטי SQL. בעת שימוש באופרטור And, הצב את כל הקריטריונים בשורה בודדת של הרשת. אם ברצונך להציג את כל הלקוחות מעיר מסוימת שהזמנתם עולה על סכום ההזמנה המינימלי, באפשרותך להתחיל בשאילתה שמסכמת מכירות לפי לקוח מעיר כלשהי. לאחר מכן, בשאילתה אחרת שמפנה לזו הראשונה, תוכל להשתמש בקריטריון מורכב בעל התניה מסוג And: קריטריון אחד לקביעת העיר

שאודותיה אתה מתעניין, וקריטריון נוסף לקביעת גבול תחתון כך שהשאילתה תחזיר פרטים על לקוחות שסכום הזמנתם גבוה מהמינימום. הפעולה מחייבת שימוש באופרטור And, ולכן שני הקריטריונים יופיעו באותה שורה של רשת עיצוב השאילתה.

לאחר מציאת כל הלקוחות העומדים בקריטריונים בעיר המבוקשת, תוכל להחליט להתמקד בדפוסי הקניה של הלקוחות באותה עיר. כדי לאסוף נתונים בנושא, תוכל להתחיל בשאילתה המצרפת את הטבלאות Orders ו- Order Details. רשת עיצוב השאילתה שלך יכולה להכיל זוגות קריטריונים נפרדים בשורות רבות. הצבת קריטריונים בשורות נפרדות ברשת עיצוב השאילתה משלבת אותם באמצעות ההתניה הלוגית Or. הקריטריונים בכל אחת מהשורות יתייחסו לעיר וללקוח. העיר תהיה משותפת לכל השורות, אך לכל שורה יהיה לקוח ייחודי משלה. כדי לקבץ מכירות בצורה אלפבתית לפי לקוח, לחץ על השורה **מיין** (Sort) של השדה Customer ובחר **סדר עולה** (Ascending).

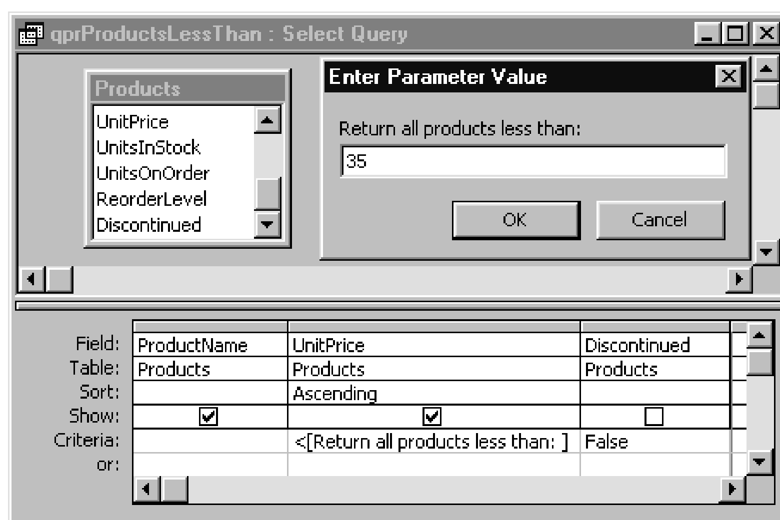
שאילתות בחירה מיוחדות

מרבית הדיון על עיצוב שאילתות עד כה התמקד במרכיבי העיצוב הבסיסיים של שאילתות בחירה (select quarries). בסעיף זה נציג שלוש הרחבות של שאילתות בחירה, אשר משפרות משמעותית את הגמישות וכושר ההתאמה של היישומים שתפתח. תחילה נלמד על אודות שאילתות פרמטר. מפתחים רבים אוהבים לעבוד עם שאילתות מסוג זה, מכיון שהן מעניקות מימד אינטראקטיבי ליישומים. אחרים מחבבים אותן עקב קלות השימוש. שאילתות איחוד (union queries) מאפשרות לעצב נתונים בצורה שלא ניתן להשיג באמצעות צירוף. כשיש צורך לשרשר שתי ערכות רשומות או יותר שיש להן שדות משותפים, שאילתות איחוד עשויות להיות הכלי היעיל ביותר. אליה וקוץ בה – שאילתות איחוד מחייבות ידע מעשי בשפת SQL. הסעיף האחרון בפרק זה מציג סקירה קצרה על התחביר של SQL ודוגמאות רבות להמחשת השימוש בשפה. נקודה נוספת: שאילתות משנה הן הכלי הנבחר כשיש צורך בקריטריונים נבונים שמתאימים את עצמם לערכים במסד הנתונים. לעיתים קרובות, שליטה בינונית או מתקדמת ב-SQL עשויה להקל על יצירת קריטריונים דינמיים שמשתנים לפי הערכים במסד הנתונים.

שאילתות פרמטר

שמן של שאילתות פרמטר נובע מהצורך לספק להן פרמטר בזמן ריצה כדי שתוכלנה להתבצע. בעת יצירת שאילתת פרמטר בתצוגת **עיצוב** (Design), תוכל לקבוע את ההנחיה להכנסת קלט בשורה **קריטריונים** (Criteria). קבע את ההנחיה במשפט הקריטריון בתור משתנה שערכו ייקבע על ידי המשתמש בזמן ריצה. ממשק המשתמש מאפשר גם להכריז על סוג נתונים לכל פרמטר קלט. פרק 2 ממחיש כיצד לקודד שאילתת פרמטר באמצעות ADO. בסעיף זה נתאר כיצד לבצע משימה זהה באמצעות ממשק המשתמש.

תרשים 4.6 מציג עיצוב של שאילתת פרמטר יחד עם תיבת הדו-שיח המוכללת של ההנחיה. שים לב שההנחיה במשפט הקריטריונים של השדה UnitePrice תחומה בסוגריים מרובעים. הסימן קטן מ (<) שלפני הסוגר הפותח מגדיר את הגבול העליון של ערכי השדה UnitePrice של קבוצת ההחזרה. אם המשתמש יקליד 35 כאשר יונחה להכניס ערך באמצעות שאילתת הפרמטר וילחץ על **אישור**, השאילתה תציג לו את רשימת כל המוצרים שמחירים נמוך מ-35 דולר. החלון התחתון בתרשים 14.6 מציג את המוצרים הראשונים ברשימה וגם את מספר הרשומות הכולל, העונים על הקריטריונים (56).



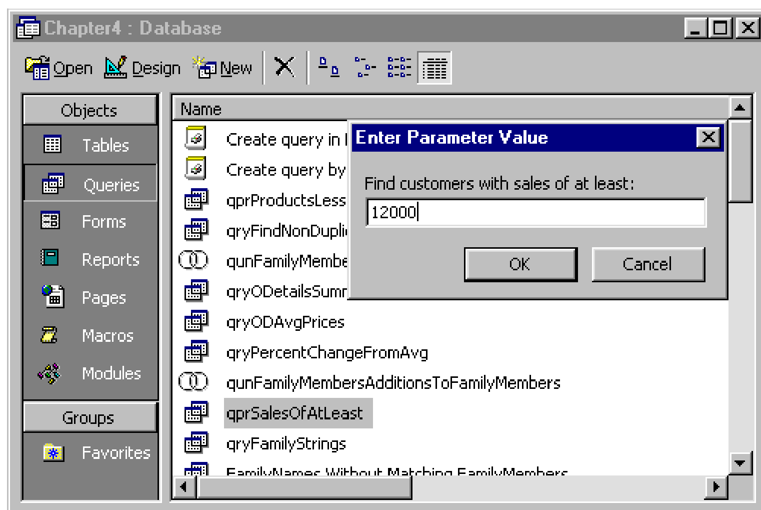
qprProductsLessThan : Select Query

Product Name	Unit Price
Geitost	$2.50
Konbu	$6.00
Filo Mix	$7.00
Tourtière	$7.45
Rhönbräu Klosterbier	$7.75
Tunnbröd	$9.00
Teatime Chocolate Biscuits	$9.20
Røgede sild	$9.50
Zaanse koeken	$9.50
Jack's New England Clam Chowder	$9.65
Sir Rodney's Scones	$10.00
Aniseed Syrup	$10.00
Longlife Tofu	$10.00
Spegesild	$12.00
Scottish Longbreads	$12.50

Record: 1 of 56

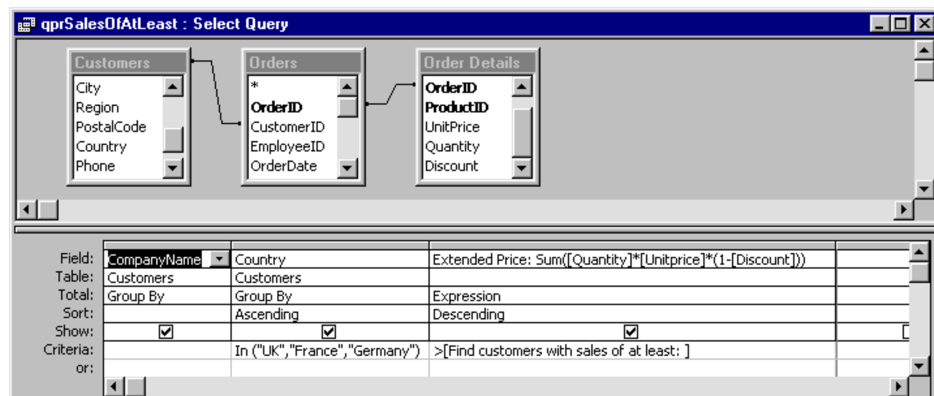
תרשים 4.16: שאילתת פרמטר וקבוצת ההחזרה שלה

משתמשי היישום לא יפעילו בדרך כלל שאילתות פרמטר מתוך תצוגת העיצוב, כפי שעשינו בתרשים 4.16. תרשים 4.17 מציג את השאילתה כפי שהיא נראית בעת הפעלתה מתוך מכולת מסד הנתונים. השאילתה בדוגמה שלפניך מחזירה את הלקוחות בבריטניה, גרמניה וצרפת, שהזמנותיהם עולות על הערך שציין המשתמש. כפי שנוכחת לדעת, אפשרות המשתמשים לקבוע את קבוצת ההחזרה הופכת את שאילתות הפרמטר לדינמיות ואינטראקטיביות.



תרשים 4.17: שאילתת פרמטר מופעלת מתוך חלון מסד הנתונים

תרשים 4.18 מציג את השאילתה מאיור 4.17 בתצוגת **עיצוב**. לשאילתה זו שני קריטריונים: האחד מנצל את מילת המפתח In כדי לבחור לקוחות שמשרדיהם הראשיים נמצאים בבריטניה, צרפת או גרמניה. כך מתבטל הצורך להגדיר שלוש שורות נפרדות שכל אחת מכילה שם מדינה אחר. ההנחיה בעמודה Extended Price ממוקמת אחרי הסימן גדול מ (>), ולכן השאילתה מחזירה את כל הרשומות שמכילות מחיר סיכום גדול מזה שמציין המשתמש.



תרשים 4.18: תצוגת העיצוב של השאילתה מתרשים 4.17

השאילתה שבתרשים 4.18 מבצעת פעולות נוספות. היא מחשבת ומסכמת את מחיר הסיכום לפי חברה וממיינת את קבוצת ההחזרה שלה בסדר אלפביתי עולה לפי מדינה ובסדר יורד לפי מחיר סיכום של ההזמנות. שיטת העיצוב גורמת להצגת הלקוחות הגדולים בראש רשימת הלקוחות בכל מדינה.

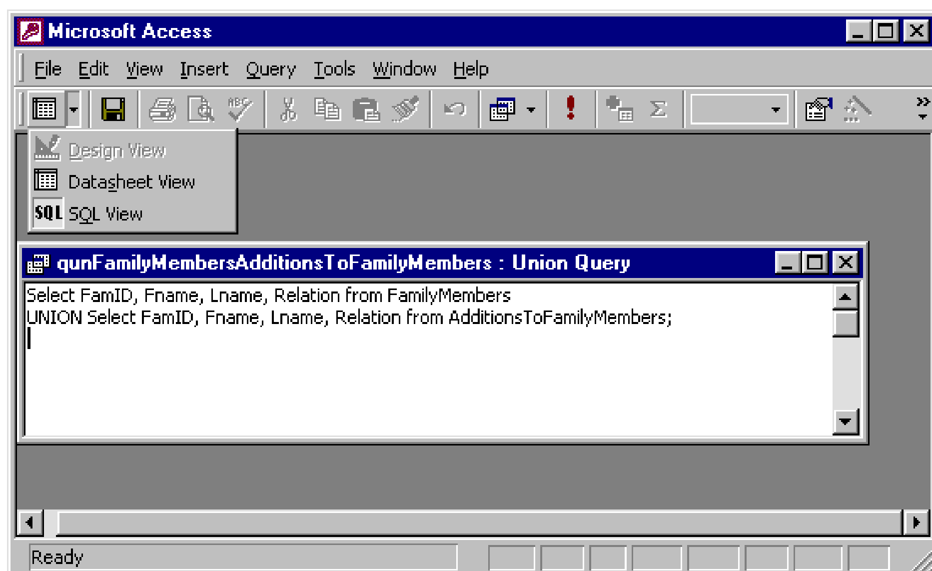
שאילתות איחוד

שאילתות איחוד הן ייחודיות מכמה היבטים. ראשית, הן מפשטות בצורה דרמטית את פעולת השרשור של שתי רשומות או יותר (כזכור, צירוף של שתי ערכות רשומות מציב אותן זו לצד זו, ולא זו אחר זו). שנית, ניתן להגדיר שאילתות איחוד בעזרת SQL בלבד. שתי התצוגות היחידות של שאילתות איחוד הן **גיליון נתונים** (Datasheet) ו-**SQL**; בשום אופן לא ניתן להציגן בתצוגת **עיצוב** (Design). שלישית, לא ניתן לעדכן ישירות את ערכי השדה בתצוגת **גיליון נתונים** של שאילתת איחוד. שאילתות בחירה רבות שנראות זהות לשאילתות איחוד בתצוגת **גיליון נתונים**, הן קבוצות רשומות דינמיות (dynasets), אשר מאפשרות למשתמש לעדכן את הטבלה שעליה מבוססת ערכת הרשומות. אם עליך לערוך את הנתונים שהם תוצאה של שאילתת איחוד, עליך ליצור עותק של הנתונים באמצעות שאילתת יצירת טבלה (make table) ולערוך אותם לאחר מכן. פעולת העריכה לא תשנה את נתוני הקלט המקוריים של שאילתת האיחוד, אך תאפשר לפחות לשנות את העותק של הנתונים שהתקבלו.

תרשים 4.19 מציג שאילתת איחוד פשוטה ביותר שמשלבת את הטבלה FamilyMembers והטבלה AdditionsToFamilyMembers. אם אתה מתקשה בעבודה עם SQL, תוכל להמשיך לשמור על פשטות שאילתות העיצוב. כתוב את המשפט הראשון בתור משפט SELECT פשוט שמציג את שמות השדות ומקור הרשומות של שדות בפסוקית FROM. ההפניות לכל שאר מקורות הרשומות צריכות להתחיל במילים UNION SELECT, אך הרשימה עדיין צריכה לכלול את שמות השדות ואת מקור הרשומות. ניתן לבצע פעולות מתקדמות יותר, אך אין הדבר הכרחי. תוכל תמיד לשלב את מקורות הרשומות בצורה פשוטה זו ולשמור את פעולות הטיפול המתוחכמות במקורות הרשומות המשולבים לשאילתות בחירה אחרת שמתבססת על הפלט של שאילתת האיחוד. באפשרותך לבצע את הפעולות המתקדמות האלו בתצוגת **עיצוב**.

ניתן לראות על פי משפטי ה- SELECT ו- UNION SELECT שאנו משרשרים טבלאות שמכילות נתונים מסוג זהה. זוהי דרישה בסיסית בעבודה עם שאילתות איחוד. מספר העמודות, המיקום, וסוג הנתונים בטבלאות, או לפחות בשדות שאתה משלב, חייבים להיות זהים. לא ניתן לעדכן את המקור מתוך תצוגת **עיצוב**, אך השאילתה מחושבת מחדש בכל פעם שפותחים אותה, עם הנתונים העדכניים שהתקבלו מהקלט שלה.

שים לב ש-Access מבטל זמינות את הסמל **תצוגת עיצוב** בשאילתת האיחוד, ולכן לא ניתן לבחון שאילתת איחוד בתצוגה זו. עצב וערוך אותה בתצוגת **SQL**, ובחן את פעולת משפטי SQL בתצוגת **גיליון נתונים**.



תרשים 4.19: שאילתת איחוד בתצוגת SQL

תרשים 4.20 מגלה את פעולת שאילתת האיחוד של תרשים 4.19. הטבלאות FamilyMembers ו-AdditionsToFamilyMembers מוצגות בחלקו השמאלי של תרשים 4.20; פלט שאילתת האיחוד הוא גיליון הנתונים בצד ימין. כפי שניתן לראות מסדר ההצגה של השדה FamID, השאילתה יוצרת גיליון נתונים חדש שמאחד את שתי השורות של הטבלה AdditionsToFamilyMembers לטבלה FamilyMembers.

FamilyMembers : Table

FamID	Fname	Lname	Relation
1	Rick	Dobson	Me
2	Virginia	Dobson	wife
3	Glen	Hill	son
4	Tony	Hill	son
5	Shelly	Hill	daughter-in-law

Record: 1 of 5

AdditionsToFamilyMembers : Table

FamID	Fname	Lname	Relation
6	Dee	Edelstein	sister
7	Foster	Simmons	nephew

Record: 1 of 2

qunFamilyMembersAdditionsToFamilyMembers ...

FamID	Fname	Lname	Relation
1	Rick	Dobson	Me
2	Virginia	Dobson	wife
3	Glen	Hill	son
4	Tony	Hill	son
5	Shelly	Hill	daughter-in-law
6	Dee	Edelstein	sister
7	Foster	Simmons	nephew

Record: 1 of 7

תרשים 4.20: שני גליונות הנתונים השמאליים הם קלט שאילתת האיחוד של תרשים 4.19. גיליון הנתונים שמימין הוא פלט שאילתת האיחוד

שאילתות משנה

שאילתת משנה היא שאילתת בחירה המשולבת בשאילתה אחרת. שאילתת המשנה חייבת להיות משפט SELECT, אך ניתן להשתמש בה בתוך שאילתה אחרת כלשהי, כגון שאילתת בחירה או שאילתת פעולה. קיימות מילות מפתח נוספות שמסוגלות למתן את פעולתה של מילת המפתח הסטנדרטית SELECT של SQL בשאילתת משנה. המשפט SELECT מחזיר ערכים, ולכן שאילתת המשנה מייצגת אפשרות חשובה לבניית קריטריונים דינמיים בשאילתות שמשתמשות בהם.

המשפט SELECT המוטבע בשורה **קריטריונים** (Criteria) של העמודה Lname בחלון התחתון של תרשים 4.4 הוא שאילתת משנה.

תרשים 4.21 מציג שאילתה שמחזירה את ערכי Lname הבלתי משוכפלים שהחזירה שאילתת איחוד שנקראת qunFamilyMembersAdditionsMoreAdditions. שאילתת האיחוד משרשרת שלושה מקורות רשומה נפרדים. החלון השמאלי התחתון בתרשים 4.21 הוא קבוצת ההחזרה של שאילתת האיחוד. שים לב להופעת ערך לא משוכפל אחד בלבד של Lname שנקרא Edelstein. החלון הימני התחתון הוא קבוצת ההחזרה של שאילתת הבחירה.

qryFindNonDuplicatedFamilyNames : Select Query

qunFamilyMembersAdditionsMoreAdditions
*
FamID
Fname
Lname
Relation

Field: Lname
Table: qunFamilyMembersAdditionsMoreAdditions
Sort:
Show: ☑
Criteria: Not In (Select Lname from qunFamilyMembersAdditionsMoreAdditions Group By Lname Having Count(*)>1)
or:

qunFamilyMembersAdditionsMoreAdditions : ...

FamID	Fname	Lname	Relation
1	Rick	Dobson	Me
2	Virginia	Dobson	wife
3	Glen	Hill	son
4	Tony	Hill	son
5	Shelly	Hill	daughter-in-law
6	Dee	Edelstein	sister
7	Foster	Simmons	nephew
8	Diane	Simmons	niece-in-law
9	Rick	Simmons	grand nephew

Record: 1 of 9

qryFindNonDuplicatedFamilyName...

Lname
Edelstein

Record: 1 of 1

תרשים 4.21: החלון העליון הוא שאילתה שמוצאת ערכי שדות לא משוכפלים במקור רשומות; שני החלונות שמתחתיו הם גליונות הנתונים של הקלט והפלט של השאילתה

החלון העליון בתרשים 4.21 הוא עיצוב שאילתת המשנה למציאת ערכי Lname לא משוכפלים. שים לב לדמיון בין שאילתה זו לשאילתה שבתרשים 4.4. לבד מן העובדה שהשאילתה סורקת מקור רשומות אחר, השוני היחיד בין השאילתות הוא ששאילתת המשנה מתחילה במילים Not In במקום במילה In. הביטוי המכיל את Not In מחזיר שמות לא משוכפלים, בעוד שהמשפט האחר מחזיר שמות משוכפלים.



הערה:
באפשרותך לשלב את קבוצות הרשומות המוחזרות משאילתות כמו אלו שבתרשימים 4.4 ו-4.21 כדי להחזיר את כל השמות השונים, אך Access כולל גם דרכים גרפיות וכאלה שמבוססות על ביטויים כדי להשיג תוצאה זהה. בהמשך הפרק נתייחס בהרחבה לגישה המבוססת על ביטויים. הגישה הגרפית כוללת מתן הערך **כן** למאפיין **ערכים ייחודיים** (unique values) של השאילתה.

שאילתות פעולה

Access כולל ארבע שאילתות פעולה שניתן להפעיל באמצעות ממשק המשתמש וגם בעזרת קוד. אלו הן שאילתות עדכון, הוספה, מחיקה ויצירה. כאיש פיתוח ברמת ביניים תיווכח ששאילתות פעולה הן מעניינות, מכיון שהן מספקות דרך מהירה וקלה יחסית לשיפור התפקודיות של יישומי Access.

באחד הסעיפים הקודמים בפרק סקרנו את שאילתת הפעולה מהיבטיה התפקודיים, מבלי להתמקד באופן היישום שלה. בסעיף זה נתאר כיצד משלבים שאילתות פעולה ביישומים מותאמים אישית. רשת עיצוב השאילתה כוללת הרחבות מיוחדות התומכות ברכיבים התפקודיים של כל אחד מסוגי שאילתת הפעולה. בסעיף זה נציג את השאילתות ואת אופן העיצוב והשימוש בהן.

שאילתות עדכון

שאילתות עדכון משמשות לשינוי ערכי שדות במקור רשומות של שאילתה. מתחילים ביצירת שאילתת עדכון בדיוק כפי שנוהגים במקרה של שאילתת בחירה. לאחר לחיצה כפולה על האפשרות **יצירת שאילתה בתצוגת עיצוב** (Create Query In Design View) בחלון **מסד נתונים** (Database), הוסף מקור רשומות אחד או יותר לשאילתה באמצעות תיבת הדו-שיח **הצגת טבלה** (Show Table). למרות שם תיבת הדו-שיח, היא מאפשרת להוסיף טבלה או שאילתה אחת או יותר. באופן טיפוסי תרצה בוודאי לצרף את כל הטבלאות והשאילתות שאתה כולל בשאילתת העדכון לאחר סגירת תיבת הדו-שיח **הצגת טבלה** (באפשרותך להסיר מקורות רשומות לא רצויים כלשהם על ידי בחירתם ולחיצה על מקש Delete במקלדת).

כדי ליצור שאילתת פעולה, עליך לבחור את הסוג שלה. בחר באפשרות **שאילתת עדכון** (Update Query) מהלחצן **סוג שאילתה** (Query Type) בסרגל הכלים **עיצוב שאילתה** (Query Design). חזות הלחצן תשתנה והוא יציג את סמל שאילתת העדכון (עיפרון ואחריו סימן קריאה). שים לב שפעולה זו מוסיפה שורה מסוג **עדכן ל** (Update To) ברשת עיצוב השאילתה ומסירה את השורות **מיין** (Sort) **והצג** (Show). לאחר מכן, הוסף שדה אחד או יותר לשורה **שדה** (Field) של רשת עיצוב השאילתה. עליך לכלול גם את השדה שברצונך לעדכן. ניתן לצרף לשאילתה שדות נוספים למטרות זיהוי או כדי לסייע להגדיר קריטריונים לקביעת הרשומות שיעודכנו.

לאחר שתסיים לבחור את השדות עבור שאילתת העדכון, הכנס ערך או ביטוי באחד התאים של השורה **עדכן ל**. הערך שבתא קובע את ערכי ההחלפה החדשים. אם לא סופקו קריטריונים כלשהם, הפעלת השאילתה תעדכן את כל הרשומות שבמקור הרשומות של השאילתה. שילוב קריטריון אחד או יותר יגביל את פעולת העדכון לשורות שעומדות בתנאים מוגדרים.

בעת הפעלת שאילתת עדכון, לא ניתן לבטל את השינויים בלחיצת עכבר. עליך לשחזר את כל הטבלה מתוך עותק גיבוי, או להפעיל שאילתת עדכון אחת או יותר כדי לשחזר את הערכים המקוריים. הלחצן **תצוגה** (View) בסרגל הכלים של תצוגת העיצוב מאפשר לבחון את הרשומות ששאילתה תשנה מבלי לבצע עדכון בפועל. לאחר שבחרת את הרשומות המתאימות, לחץ על לחצן **הפעל** (Run) כדי להפעיל את השאילתה. תיפתח תיבת הודעה ובה מספר הרשומות שהשאילתה תעדכן. באפשרותך לבטל את שאילתת העדכון בשלב זה על ידי לחיצה על **לא**. אם תלחץ על **כן**, יימשך תהליך העדכון. אם השדה אינו בר-עדכון, Access מציג תיבת הודעה מתאימה.

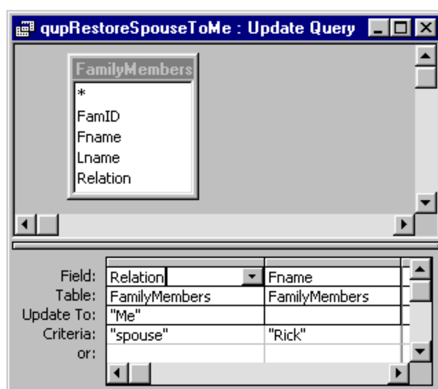
Access מציג תיבת הודעה נוספת אם מנסים להפעיל את שאילתת העדכון מחוץ לתצוגת העיצוב. באפשרותך לדחות את האפשרות לשנות את ערכי השדות או להמשיך. אם תבחר להמשיך בעדכון, תיבת הדו-שיח הבאה דומה לזו הראשונה שהוצגה בתצוגת העיצוב. התיבה מציינת את מספר הרשומות שישתנו ומאפשרת לשנות את הרשומות פעם נוספת ואחרונה.

שאילתות עדכון יכולות לשנות שדות רבים במקורות הרשומות שלהם. באפשרותך לעדכן את השדות במקור רשומות כל עוד בידך הרשאה לעשות זאת ומקור הרשומות זמין. נתונים מקשר גומלין מסוג יחיד לרבים הם לעיתים קרובות ניתנים לעדכון, אך לא תמיד. לדוגמה, לא ניתן לשנות את המפתח הראשי, אם עדכונים על פי היררכית קשרים אינם ניתנים לשימוש. שדות וערכים מחושבים בשאילתות הצלבות (crosstab) אף הם אינם ניתנים לעדכון. אם שדה מחושב, שאילתות פעולה אינן יכולות לשנות את ערכיו. העזרה המקוונת כוללת תיאורים מפורטים של שדות שאינם ניתנים לעדכון.

החלון העליון בתרשים 4.22 מציג את רשת עיצוב השאילתה של שאילתת עדכון בסיסית שמשנה שני ערכי שדה בטבלה FamilyMembers. שאילתה זו משנה את ערך השדה Relation ל-spouse, אם הערך הקודם בשדה מכיל את הערך me או wife. שני החלונות התחתונים בתרשים מציגים שתי שאילתות עדכון נוספות שמשחזרות את הערכים המקוריים של מקור הרשומות. שאילתות אלו מציינות גישה אחת לבניית

שאילתות עדכון, כך שבאפשרותך לשחזר ערכים התחלתיים. שים לב כי יש צורך בשתי שאילתות כדי לבטל את פעולת השאילתה המקורית. כמו-כן, שים לב שכל שלושת המקרים מכילים קריטריונים מורכבים. הקריטריונים המורכבים מסייעים לזהות טוב יותר את הרשומות המיועדות לעדכון.







תרשים 4.22: שלוש שאילתות עדכון

הפעלת שאילתות השחזור טומנת בחובה כמה בעיות. ראשית, עליך להפעיל את שתיהן. שנית, Access מבקש את אישורך להפעלת כל שאילתה (אם תפעיל אותן מתוך החלון **מסד נתונים**, הוא מבקש את האישור פעמיים). אם ערכת עדכונים מסוגים רבים, כדאי יהיה להפוך את התהליך לאוטומטי כדי לחסוך זמן ולשפר את הדיוק. שגרת המשנה שלפניך מבצעת את פעולת העדכון. כל שעל המשתמש לעשות הוא להפעיל אותה.

```
Sub RestoreMeAndWife()
   DoCmd.SetWarnings False
   DoCmd.OpenQuery "qupRestoreSpouseToMe"
   DoCmd.OpenQuery "qupRestoreSpouseToWife"
   DoCmd.SetWarnings True
End Sub
```

הקוד של שתי שאילתות העדכון מכיל ארבע שורות. השיטה SetWarnings מבטלת את תיבות הדו-שיח של האזהרה. השיטה OpenQuery משמשת להפעלת שתי השאילתות, כל אחת בתורה. השורה האחרונה מחדשת את השימוש בתיבות הדו-שיח של האזהרה.

שאילתות הוספה

שאילתות הוספה מוסיפות רשומות חדשות מתוך ערכת רשומות של מקור אל ערכת רשומות יעד. ערכת רשומות היעד יכולה להיות טבלה בודדת או טבלאות שעליהן מבוססת שאילתת בחירה ברת-עדכון. ערכת רשומות המקור יכולה להיות טבלה או שאילתת בחירה בעלת שדות שלפחות חלק מהם תואמים את השדות של ערכת רשומות יעד. ערכת רשומות המקור חייבת להיות במסד הנתונים הנוכחי. ערכת רשומות היעד יכולה להיות בקובץ מסד נתונים כלשהו של Access.

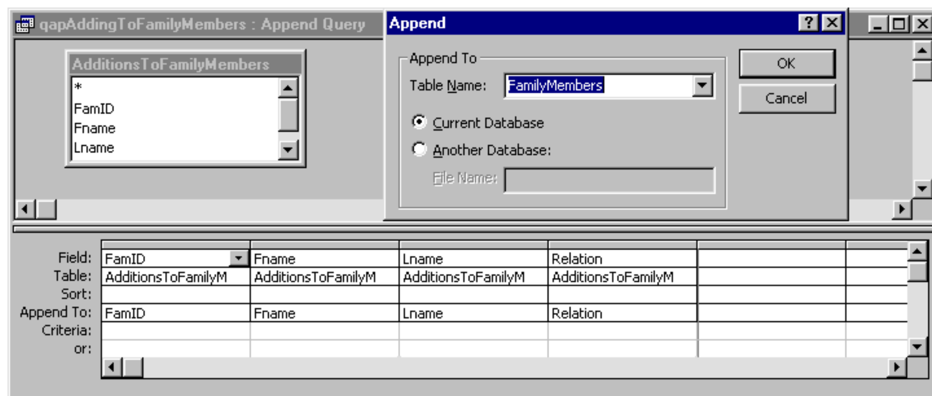
שאילתת הוספה אינה חייבת להוסיף רשומות המכילות כל שדה ושדה ביעד, אך עליה להכיל שדה מבוקש אחד לפחות. אם טבלת היעד מכילה מפתח ראשי, ודא שערכי המפתח ייחודיים בשתי ערכות הרשומות. אם לא, ערכי המפתח הראשי לא יהיו ייחודיים, שאילתת ההוספה לא תצליח להוסיף רשומות.

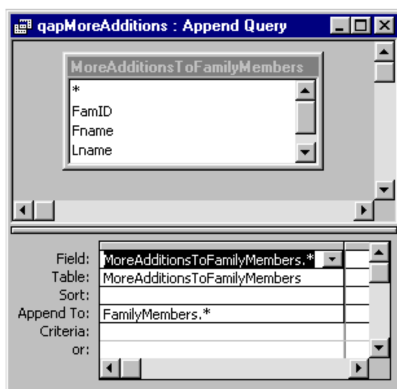
מתחילים בעיצוב שאילתת הוספה על ידי הוספת ערכת רשומות המקור לתצוגת העיצוב של השאילתה. לאחר מכן, מוסיפים שדות מתוך המקור אל השורה **שדה** (Field) של רשת עיצוב השאילתה. לאחר מכן, פתח את הלחצן **סוג שאילתה** (Query Type) בסרגל הכלים ובחר את **שאילתת הוספה** (Append Query) כדי להציג את תיבת הדו-שיח **הוספה** (Append). תיבה זו מפשטת את משימת העיצוב של טבלה אחרת במסד הנתונים הנוכחי בתור טבלת היעד. באפשרותך גם לציין שברצונך להוסיף רשומות לטבלה בקובץ מסד נתונים אחר של Access על ידי בחירת לחצן הבחירה **מסד נתונים אחר** (Another Database) והקלדת הנתיב ושם קובץ של מסד הנתונים המבוקש. לאחר בחירת מסד נתונים, תיבת הרשימה הנפתחת **שם הטבלה** (Table Name) מציגה את הטבלאות בקובץ שבחרת. ציין ערכת רשומות יעד על ידי בחירת שם טבלה מתוך תיבת הרשימה. אם ברצונך להוסיף רשומות לשאילתת בחירה, עליך ליצור בעצמך את קוד שאילתת הפעולה (עיין בפרק 2 לקבלת דוגמאות לביצוע פעולה זו באמצעות ADO ו-SQL).

לאחר בחירת מקור רשומות יעד, Access משנה את רשת עיצוב השאילתה שלו כך שתכלול את השורה **הוסף ל** (Append To). הוא גם מתאים אוטומטית שדות רבים ככל שהוא מזהה בערכת רשומות היעד לאלה שבערכת רשומות המקור. ניתן לדרוס את הבחירות האוטומטיות. שדות ש-Access אינו מתאים להם בני זוג, תוכל להתאים בעצמך (לדוגמה, שדה ששמו שונה, אך סוג הנתונים שלו זהה). לחץ בשורה **הוסף ל** של עמודה, ובחר בשם השדה המתאים מתוך הרשימה הנפתחת של שמות שדות ערכת רשומות היעד.

החלון העליון בתרשים 4.23 הוא שאילתת הוספה בתצוגת עיצוב לאחר הוספת שדות בשורות **שדה** ו**הוסף ל** שלה, ותיבת הדו-שיח **הוספה** שלה פתוחה. שאילתת הוספה זו

מוסיפה רשומות מתוך הטבלה AdditionsToFamilyMembers. לאחר סגירת תיבת הדו-שיח **הוספה**, לחץ על הלחצן **הפעל** (Run) כדי להפעיל את השאילתה. בדומה לשאילתת פעולה אחרת כלשהי, לחץ על לחצן **תצוגה** (View) בסרגל הכלים כדי להציג את הרשומות שהשאילתה תעבד מבלי להפעיל את השאילתה.





תרשים 4.23: החלון העליון הוא שאילתת הוספה שמציגה את כל השדות שיש להוסיף למקור רשומות היעד. החלון התחתון הוא שאילתת הוספה שמנצל את הכוכביות כדי לייעד את כל השדות של ערכות הרשומות היעד והמקור.

החלון התחתון של תרשים 4.23 מציג גירסה שונה של שאילתת ההוספה שבחלון העליון. השאילתה התחתונה מעתיקה את כל השדות מהטבלה MoreAdditionsToFamilyMembers אל הטבלה FamilyMembers. גררתי את הכוכבית מרשימת שדה ערכת רשומות המקור לשורה **שדה** (Field) שברשת עיצוב השאילתה. באופן דומה, בחרתי את הכוכבית מהרשימה הנפתחת בשורה **הוסף ל**. שתי השאילתות מוסיפות את כל השדות הקיימים מערכת רשומות המקור אל ערכת רשומות היעד. לעומת זאת, השאילתה התחתונה גמישה יותר, כיון שהיא כוללת אוטומטית שדות חדשים כלשהם שתוסיף לשתי הטבלאות. כאשר מציגים את השדות ברשימה באופן מפורש, השאילתה מוגבלת לשדות אלה עד שמעדכנים ידנית את הערכים בשורות **שדה** ו**הוסף ל**.



הערה:

במקרים אחדים, Access עלול לא לרענן מיידית את התצוגה של טבלה לאחר השלמת שאילתת הוספה. לכן, השאילתה יכולה לעדכן בפועל את ערכת רשומות היעד מבלי שהשינוי יבוא לידי ביטוי בתצוגה. ניתן להתגבר על הבעיה על ידי שינוי תצוגת ערכת רשומות היעד לתצוגת **עיצוב** ולחזור מייד לתצוגת **גיליון נתונים** (לחילופין ניתן לרענן את תצוגת ערכת רשומות, על ידי הקשה על Shift+F9).

שאילתות מחיקה

שאילתות מחיקה מסירות רשומות באופן תמידי מערכת רשומות יעד. מסיבה זו עליך לנהוג זהירות רבה בעת השימוש בשאילתות פעולה למחיקה. הקפד להשתמש בלחצן **תצוגה** (View) כדי לוודא שהרשומות המיועדות להסרה הן אכן אלה שברצונך להסיר בטרם תפעיל את השאילתה. שאילתות מחיקה יכולות למחוק גם רשומות בערכות רשומות אחרות פרט ליעד. הדבר נכון במצב שהיעד נמצא בצד היחיד של קשר הגומלין מסוג יחיד לרבים ושנבחרה האפשרות למחיקות בהתאם להיררכית הקשרים. במצב זה, מחיקת רשומה בצד היחיד מסירה את הרשומות התואמות גם בצד הרבים.

כללית, עליך להגן על מקורות הנתונים בעת השימוש בשאילתות פעולות, אך הדבר נכון במיוחד לגבי שאילתות מחיקה, על שום יכולתן למחוק קבוצות גדולות של רשומות מתוך מקור רשומות. מומלץ לשקול הכנת עותק גיבוי של מקור הרשומות, וכך תוכל לשחזר רשומות שנמחקו בשוגג. אסטרטגיה נוספת בטיפול בשאילתות מחיקה היא להסתיר אותן בחלון **מסד נתונים** (Database), מה שיצמצם את הסיכוי שהמשתמש יפעיל בטעות שאילתת מחיקה (באפשרותך להפעיל באמצעות קוד את שאילתות המחיקה שבחרת להסתיר). אובייקטים מוסתרים של מסד נתונים אינם מוצגים אלא רק כשתבחר להציגם כך: בחר באפשרות **אפשרויות** (Options) מהתפריט **כלים** (Tools) ובחר את **אובייקטים מוסתרים** (Hidden Objects) בכרטיסיה **תצוגה** (View) של תיבת הדו-שיח **אפשרויות**.



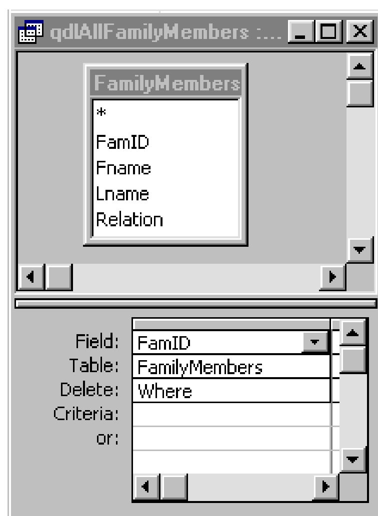
הערה:

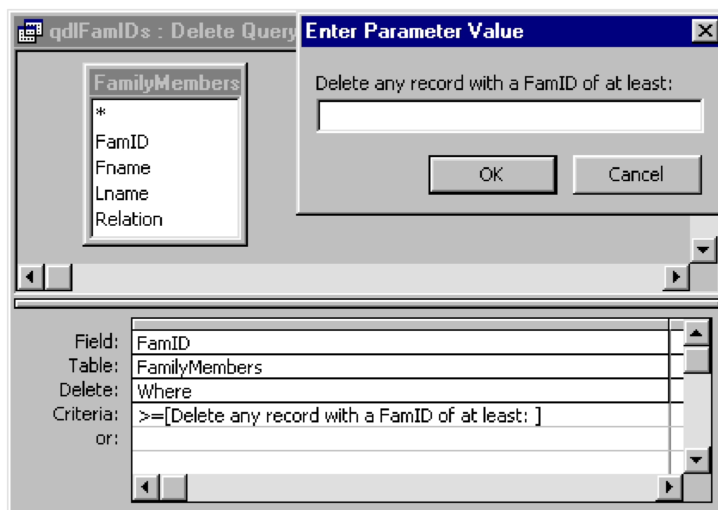
להסתרת שאילתת פעולה, לחץ לחיצה ימנית על שאילתה בחלון **מסד נתונים** ובחר באפשרות **מאפיינים** (Properties) מתוך תפריט הקיצור. בחר את תיבת הסימון **מוסתר** (Hidden) בגיליון המאפיינים ולחץ על **אישור** (ייתכן שתצטרך לבטל את גם בחירת תיבת הסימון **אובייקטים מוסתרים** בכרטיסיה **תצוגה** של תיבת הדו-שיח **אפשרויות**).

הפעולה הראשונה בעיצוב שאילתת מחיקה היא לחיצה כפולה על **יצירת שאילתה בתצוגת עיצוב** (Create Query In Design View) בחלון **מסד נתונים**. בתיבת הדו-שיח **הצגת טבלה** (Show Table), הוסף את מקור הרשומות שממנו ברצונך למחוק רשומות. ציין את השורות שברצונך למחוק על ידי גרירת שדות מתוך רשימת השדות אל השורה **שדה** (Field) ברשת עיצוב השאילתה. הכנס ערכים בשורה **קריטריונים** (Criteria) של

רשת העיצוב, כדי לציין רשומות יעד המיועדות למחיקה. אם ברצונך ששאילתת המחיקה תסיר את כל הרשומות ממקור רשומות היעד, הכנס כוכבית (או שם שדה כלשהו) בשורה **שדה** ואל תכניס קריטריון כלשהו. השורה **שדה** חייבת להכיל ערך אחד לפחות, כדי ששאילתת המחיקה תוכל לפעול.

תרשים 4.24 מציג את העיצוב של זוג שאילתות מחיקה. השאילתה העליונה מוחקת את כל הרשומות מתוך הטבלה FamilyMembers; היא אינה מכילה ערכים כלשהם בשורה **קריטריונים**. לו ידעת את ערכי FamID של הרשומות הספציפיות שברצונך למחוק, יכולת להוסיפם לשורות **קריטריונים** טרם הפעלת השאילתה.





תרשים 4.24: שתי שאילתות פרמטר

אם אינך יודע איזה שורות למחוק בזמן העיצוב, באפשרותך ליצור את שאילתת המחיקה בתור שאילתת פרמטר. החלון התחתון בתרשים 4.24 מציג סוג שאילתה זה. ההנחיה היא ההנחיה של שאילתת פרמטר שמופיעה בזמן ריצה. משתמש יכול להכניס ערך FamID כדי למחוק את כל הרשומות שערך FamID שלהן גדול או שווה לערך שצוין.

השיגרה שלפניך שופכת מעט אור על שאילתות מחיקה והוספה יחדיו. השיגרה מאפשרת למחוק או לשחזר את כל הרשומות בטבלה FamilyMembers. כשהיא מסתיימת, הטבלה פתוחה וזמינה להצגה. אם תסתיר את השאילתות qdlAllFamilyMembers ו-qapRestoreAllFamilyMember, הדרך היחידה להפעיל אותן תהיה באמצעות קוד, כמתואר להלן.

```
Sub ActionQueryDemo()
Dim varYesNo

' Prompt to delete all records.
    varYesNo = MsgBox("Do you want to remove all records?", _
        vbYesNo, "Programming Microsoft Access 2000")
' If answer is yes, remove them all.
    If varYesNo = vbYes Then
        DoCmd.SetWarnings False
        DoCmd.OpenQuery "qdlAllFamilyMembers"
        DoCmd.SetWarnings True
    End If
' Prompt to restore all records.
    varYesNo = MsgBox("Do you want to restore all records?", _
        vbYesNo, "Programming Microsoft Access 2000")

' If answer is yes, restore them all.
    If varYesNo = vbYes Then
        DoCmd.SetWarnings False
        DoCmd.OpenQuery "qapRestoreAllFamilyMembers"
        DoCmd.SetWarnings True
    End If
' Closing and opening table redraws it while making the table available for viewing.
DoCmd.Close acTable, "FamilyMembers"
DoCmd.OpenTable "FamilyMembers"
End Sub
```

השיגרה פותחת בפונקציה מסוג MsgBox ששואלת את המשתמש אם ברצונו להסיר את כל הרשומות מהטבלה FamilyMembers. אם המשתמש עונה בחיוב, השיגרה מפעילה את השאילתה qdlAllFamilyMembers שמופיעה בחלק העליון של תרשים 4.24 (שים לב כי ActionQueryDemo מכילה זוג משפטים שמכבים תיבות דו-שיח של אזהרה ולאחר מכן מפעילים אותן שוב). לאחר מכן, השיגרה שואלת את המשתמש אם ברצונו לשחזר

את כל הרשומות של FamilyMembers. אם המשתמש עונה בחיוב, הקוד מפעיל את השאילתה qapRestoreAllFamilyMember כדי להוסיף רשומות מתוך עותק גיבוי של הטבלה אל הטבלה FamilyMembers.

לפני סיומה, השיגרה סוגרת את הטבלה FamilyMembers ופותחת אותה מחדש, פעולה שמבטיחה שהטבלה תציג את הנתונים העדכניים ביותר, ומשאירה אותה פתוחה לצפיה ב-Access.

שאילתות יצירת טבלה

שאילתות יצירת טבלה, בניגוד לשלוש שאילתות הפעולה האחרות, אינן משנות, מוסיפות או מוחקות מקור נתונים מקורי. המטרה היחידה של שאילתת יצירת טבלה היא ליצור טבלה של Access המבוססת על מקור רשומות. שילוב כלשהו של נתונים שניתן לקשר בשאילתת בחירה, ייחשב למקור רשומות חוקי שממנו ניתן ליצור טבלה.

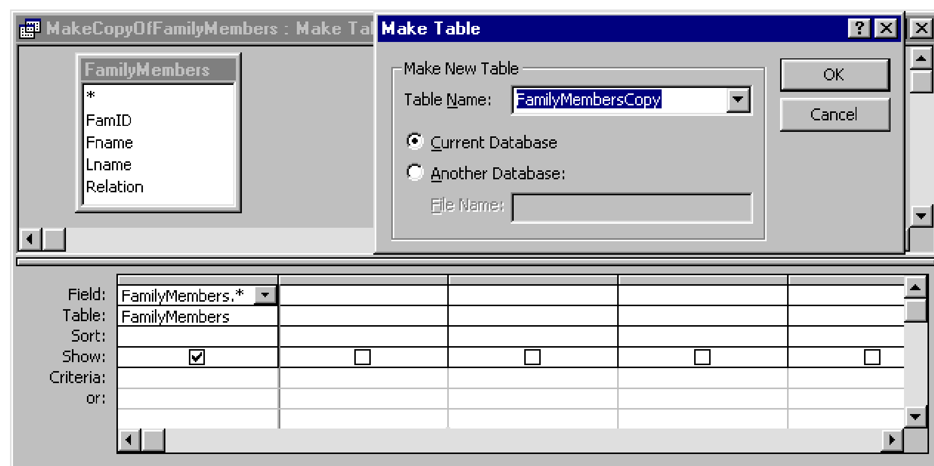
תרחישי יישום אחדים עשויים להצריך שאילתות יצירת טבלה. ראשית, באפשרותך להשתמש בשאילתת יצירת טבלה כדי ליצור עותקי גיבוי של טבלאות קריטיות. הדבר נכון במיוחד רגע לפני שיש למחוק רשומות מתוך מאגר הנתונים של המערכת הפעילה. שנית, שאילתת יצירת טבלה יכולה לשמר את מצב הטבלה או מצב קבוצת טבלאות מצורפות בנקודת זמן קבועה. שלישית, ניתן להכין עותק נתונים בקובץ מסד נתונים אחד למען המטרות של יישום מסד נתונים אחר. תמיד קיימת אפשרות לקשר נתונים (היתרון של קישור הוא נתונים עדכניים יותר), אך ניתן לשפר את הביצועים באמצעות טבלה בקובץ, במקום טבלה מקושרת שמקורה בקובץ אחר. רביעית, שאילתות עלולות להפוך למורכבות מדי לעיבוד ב-Access, או שעיבודן עלול להימשך זמן רב. במצבים כאלה, החלף את השאילתה המורכבת הבעייתית בטבלה שנוצרה באמצעות שאילתת יצירת טבלה. על ידי כך תעקוף את הודעת השגיאה הנוגעת למורכבות יתר וביצועי השאילתה ישתפרו.

שאילתת יצירת טבלה היא במהותה שאילתת פעולה, ולכן קל בהרבה ליצור טבלה באמצעות שאילתת יצירת טבלה לעומת יצירת טבלה בעזרת ADO, SQL או DAO. כך יוצרים שאילתת יצירת טבלה: פתח שאילתה ריקה בתצוגת **עיצוב**, הוסף לה טבלאות ושאילתות בתור משאבים לטבלה החדשה, צרף קלטים לפי הצורך ובחר באפשרות **שאילתת יצירת טבלה** (Make-Table Query) מתוך רשימת הלחצן **סוג שאילתה** (Query Type). כעת, גרור את השדות שתזדקק להם בטבלה החדשה אל השורה **שדה** (Field) של רשת עיצוב השאילתה. לחץ על לחצן **הפעל** (Run) כדי ליצור את הטבלה החדשה. אם קיימת כבר טבלה הנושאת את שמה של טבלת היעד, Access ישאל אם ברצונך למחוק את הגירסה הישנה בטרם ייצור טבלה חדשה.

תרשים 4.25 מציג שאילתת יצירת טבלה פשוטה שיוצרת עותק גיבוי של הטבלה FamilyMembers. תיבת הדו-שיח **יצירת טבלה** (Make Table) מופיעה בעת הפעלת שאילתת יצירת טבלה. באפשרותך לנצל את תיבת הדו-שיח כדי להגדיר שם טבלה

שנמצאת בקובץ מסד הנתונים הנוכחי, או נתיב ושם קובץ במסד נתונים אחר. לאחר מכן תוכל לבחור טבלה להעתקה מתוך תיבת הרשימה הנפתחת **שם הטבלה** (Table Name).

בדוגמה שבתרשים 4.25, הטבלה FamilyMembersCopy שנמצאת במסד הנתונים הנוכחי מוגדרת בתור יעד מסד הנתונים. הערך בשורה **שדה** (Field) שברשת עיצוב השאילתה מציין כי כל השדות בטבלה FamilyMembers לוקחים חלק בטבלה החדשה. לא מצוינים קריטריונים כלשהם להכללה או להשמטה של שורות מערכת רשומות המקור.



תרשים 4.25: שאילתת יצירת טבלה פשוטה

תכנות שאילתות באמצעות SQL ו-ADO

SQL היא שפה מקובלת ביותר לטיפול בנתונים יחסיים. רוב המנגנונים לניהול מסדי נתונים יחסיים מבוססים על SQL, וכמעט כל שאילתה שיוצרים בתצוגת העיצוב ניתן ליצור באמצעות קוד SQL. אם יש לך ידע מעשי כללי ב-SQL (שנקראת גם SQL-92), תוכל לעצב משפטי שאילתה יעילים עבור השאילתות שלך בשפה הטבעית של מנגנוני מסדי הנתונים הללו.

הסעיפים שלפניך מציגים שימושים מקובלים של SQL ביישומי Access. ראשית, נבחן את משפטי SELECT. לאחר מכן נלמד על פונקציות להגדרת נתונים. משפטי SQL המיישמים הגדרת נתונים יכולים להתבסס ישירות על התחביר של משפט SELECT ולשכלל את רמת התפקודיות שלו. לבסוף, נסקור טכניקות SQL ליצירה והפעלה של תצוגות ושגרות מאוחסנות.

משפטי SELECT

משפט SELECT של SQL מחזיר קבוצת שורות מתוך טבלה אחת או יותר של מסד נתונים. באפשרותך להשתמש בפסוקיות קשורות, כגון WHERE ו- ORDER BY כדי לשכלל בצורה דרמטית את האופן שמשפט SELECT בוחר ומציג שורות בקבוצת ההחזרה שלו. פונקציות צבירה של SQL כוללות אפשרויות נוספות לניהול קבוצת ההחזרה.

משפט SELECT בסיסי מציין עמודה אחת או יותר מתוך מקור רשומות. הנה התבנית הבסיסית שלו:

```
SELECT FieldList FROM RecordSource
```

רשימות שדות

בוחרים קבוצת משנה של שדות במקור רשומות על ידי הגדרת רשימה מופרדת בפסיקים עבור העמודה FieldList. בחירת כל השדות במקור רשומות מתבצעת באמצעות כוכבית (*). אם שדה אחד או יותר בעמודה FieldList נקראים בשם זהה, אך נמצאים בטבלאות שונות, עליך להקדים את שמות הטבלאות לשמות השדות ולהפריד ביניהם באמצעות נקודה. אם שם הטבלה מכיל תו רווח אחד או יותר, עליך להכניס את שם הטבלה בין סוגריים.

צירופים

כדי לציין את קשר הגומלין בין שתי טבלאות המשמשות כמקור רשומות עבור קבוצת רשומות המוחזרת ממשפט SELECT, אתה משתמש ב**צירופים** (joins). הפסוקית FROM של משפט SELECT מציינת צירופים כלשהם לשילוב שתי טבלאות או יותר. צירוף חייב לציין לפחות את העמודות שישמשו לשילוב טבלאות ואת אופן קביעת ההתאמה בין הטבלאות (ניתן לקנן צירופים כדי לציין קשרי גומלין בין יותר משתי טבלאות). סוגי הצירופים הטיפוסיים הם צירופים פנימיים, שבוחרים רשומות תואמות מתוך כל טבלה בזוג טבלאות; צירופים שמאליים (left joins) שכוללים את כל הרשומות מתוך הטבלה הראשונה ורק רשומות תואמות מהטבלה השנייה; וצירופים ימניים (right joins) שכוללים את כל הרשומות מתוך הטבלה השנייה ורק רשומות תואמות מהטבלה הראשונה.

פסוקיות WHERE

משתמשים בפסוקית WHERE של משפט SELECT להגדרת קריטריונים לבחירת שורות מתוך מקור רשומות. בניגוד לפסוקית FROM, הפסוקית WHERE היא אופציונלית ומשתמשים בה רק כדי להגביל את הרשומות בקבוצת החזרה.

האופרטורים LIKE

האופרטור האופציונלי LIKE משמש למציאת ערכים בשדות שתואמים לתבנית (pattern) מסוימת. אופרטור זה תומך בהתאמת תבנית בין המנגנונים של מסדי הנתונים Jet ו- SQL Server, אך עליך לנהוג בזהירות, מכיון שהמנגנון של שרת קבצים Jet מפרש תווים כלליים (wildcard characters) באופן שונה מזה שעושים מנגנוני שרת-לקוח רבים. לדוגמה, Microsoft SQL Server משתמש בתווים הכלליים % ו- ^ לעומת Jet שמשתמש בתווים * ו-?.

פסוקיות ORDER BY

הפסוקית האופציונלית ORDER BY משמשת במשפט SELECT כדי להחזיר ערכת רשומות בסדר שונה מזה שמכתיב המפתח הראשי. מציינים את השדה הראשון שלפיו רוצים למיין מייד לאחר מילת המפתח ORDER BY. ברירת המיון של סדר המיון היא עולה (ASC), אך באפשרותך לציין גם סדר יורד (DESC). אם ברצונך למיין לפי יותר משדה אחד, הפרד בין כל צמד של שם שדה וסדר מיון, באמצעות פסיק.

הפסוקיות GROUP BY ו-HAVING

פסוקיות אלו משלימות זו את זו. הן פועלות עם פונקציות צבירה SQL כגון COUNT ו-SUM. הפסוקית GROUP BY מציינת שדות שלפיהן יש לחשב צבירים (aggregates). הפסוקית HAVING דומה לפסוקית WHERE; היא מגבילה את ערכת הרשומות המוחזרת באמצעות הפסוקית GROUP BY לרשומות שתואמות לקריטריונים נתונים.

מילות המפתח DISTINCT

מילת המפתח DISTINCT היא אחת הדרכים להגבלת מספר הרשומות שמוחזרות באמצעות משפט SELECT. מילת המפתח מונעת כפילות נתונים בשדות שנבחרו. הצב את מילות המפתח DISTINCT בין מילת המפתח SELECT לבין רשימת השדות. התבנית הכללית של מילת המפתח:

```
Select Distinct FieldList from Recordset
```

שימוש באובייקט Command לביצוע SQL

האובייקט Command של ADODB משמש לביצוע פקודת SQL כנגד מקור נתונים באופן הבא:

1. הצב במאפיין CommandText של האובייקט Command את משפט ה-SQL שברצונך לבצע.
2. הגדר את המאפיין CommandType של האובייקט בתור adCmdText כדי למטב ביצועים.

3. הפעל את השיטה Execute של האובייקט Command כדי ליצור את קבוצת ההחזרה של SQL.

4. פתח אובייקט Recordset המבוסס על האובייקט Command כך שתוכל לטפל בקבוצת ההחזרה של SQL ביישום שלך.

הקוד שלפניך מנצל את האובייקט Command כדי לבצע משפט SQL פשוט:

```
Sub MySelect()
Dim cnn1 As New ADODB.Connection
Dim cmd1 As ADODB.Command
Dim rst1 As ADODB.Recordset
' Create the connection to another database.
   cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
   "Data Source=C:\Program Files\Microsoft Office\Office\" & _
   "Samples\Northwind.mdb;"
' Define and execute command to select all ProductID field
' values from a single table.
   Set cmd1 = New ADODB.Command
   With cmd1
      .ActiveConnection = cnn1
      .CommandText = "Select ProductID from [Order Details]"
      .CommandType = adCmdText
      .Execute
   End With
' Assign the return set to a recordset.
   Set rst1 = New ADODB.Recordset
   rst1.CursorType = adOpenStatic
   rst1.LockType = adLockReadOnly
   rst1.Open cmd1
   Debug.Print rst1.RecordCount
End Sub
```

השיגרה MySelect מדפיסה את מספר הרשומות בקבוצת ההחזרה (2155) בחלון Immediate (מיידי). קבוצת ההחזרה מתוך משפט SQL כוללת כל רשומה בטבלה Order Details. כברירת מחדל, המשפט SELECT בוחר את כל הרשומות שבמקור הרשומות שברקע, ומשפט SQL של השיגרה MySelect אינו מכיל מגבלה כלשהי לגבי הרשומות שיוחזרו.

דוגמה באמצעות INNER JOIN

המשפט SELECT שלפניך ממחיש רכיבים חדשים אחדים. ראשית, הוא מציג את התחביר של צירוף פנימי בין הטבלאות Order Details ו-Products. שנית, הוא ממספר את השורות בקבוצת ההחזרה ומדפיס אותן בחלון Immediate.

```
Sub MySelect3()
Dim cnn1 As New ADODB.Connection
Dim cmd1 As ADODB.Command
Dim rst1 As ADODB.Recordset, int1 As Integer

' Create the connection to another database.
    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
    "Data Source=C:\Program Files\Microsoft Office\Office\" & _
    "Samples\Northwind.mdb;"

' Define and execute command to select distinct ProductName field
' values from a pair of joined tables.
    Set cmd1 = New ADODB.Command
    With cmd1
        .ActiveConnection = cnn1
        .CommandText = "Select Distinct ProductName from " & _
            "[Order Details] Inner Join Products on " & _
            "[Order Details].ProductID = Products.ProductID"
        .CommandType = adCmdText
        .Execute
    End With

' Assign the return set to a recordset
' and print the results to the Immediate window.
    Set rst1 = New ADODB.Recordset
    rst1.CursorType = adOpenStatic
    rst1.LockType = adLockReadOnly
    rst1.Open cmd1
    Debug.Print rst1.RecordCount
    For int1 = 1 To rst1.RecordCount
        Debug.Print rst1("ProductName")
        rst1.MoveNext
    Next int1
End Sub
```

המשפט SELECT של השיגרה MySelect3 מתבסס על שדות מתוך שתי טבלאות, ולא אחת בלבד, כמו בדוגמה הקודמת. המשפט מבצע צירוף פנימי על הטבלה Products ועל הטבלה Order Details כדי להתאים בין השדות ProductID של שתי הטבלאות. מאחר שמשפט SQL משתמש במילת המפתח DISTINCT, לא תהיה החזרה של ערכים כפולים מהשדה ProductName. לבסוף, הצירוף הפנימי מאפשר לשיגרה להדפיס את השם התיאורי של המוצר שמתאים לכל קוד זיהוי של מוצר.

דוגמה לשימוש ב- SUM ו- ORDER BY

השיגרה שלפניך משתמשת בפונקציית הצבירה SUM ובאפשרויות הצירוף והמיון שלה. השיגרה מחשבת את סכום מחיר הסיכום לשורה לכל מוצר בטבלה Order Details. קבוצת ההחזרה של המשפט ממיינת את התוצאות לפי גובה ההכנסה שמניב כל מוצר (מהגבוהה עד לנמוכה ביותר).

```
Sub MySelect4()
Dim cnn1 As New ADODB.Connection
Dim cmd1 As ADODB.Command
Dim rst1 As ADODB.Recordset, int1 As Integer
' Create the connection to another database.
    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
    "Data Source=C:\Program Files\Microsoft Office\Office\" & _
    "Samples\Northwind.mdb;"
' Define and execute command to select distinct ProductName field
' values from a pair of joined tables; compute extended price.
    Set cmd1 = New ADODB.Command
    With cmd1
        .ActiveConnection = cnn1
        .CommandText = "Select Distinct Products.ProductName, " & _
            "Sum([Order Details].[UnitPrice]*" & _
            "[Order Details].[Quantity]*" & _
            "(1-[Order Details].[Discount])) As [Extended Price] " & _
            "From Products Inner Join [Order Details] On " & _
            "Products.ProductID = [Order Details].ProductID " & _
            "Group By Products.ProductName " & _
            "Order By Sum([Order Details].[UnitPrice]*" & _
            "[Order Details].[Quantity]*" & _
            "(1-[Order Details].[Discount])) Desc"
        .CommandType = adCmdText
        .Execute
    End With
' Assign Select statement return set to a recordset
' and print the results to the Immediate window.
    Set rst1 = New ADODB.Recordset
    rst1.CursorType = adOpenKeyset
    rst1.Open cmd1
    Debug.Print rst1.RecordCount
    For int1 = 1 To rst1.RecordCount
        Debug.Print rst1("ProductName"), rst1.Fields("Extended Price")
        rst1.MoveNext
    Next int1
End Sub
```

השיגרה MySelect4 יוצרת שדה מחושב באמצעות קוד. השיגרה מפעילה את הפונקציה SUM עם הפסוקית GROUP BY כדי לחשב את ההכנסה שיצר כל אחד מהמוצרים. לולא פסוקית זו, הקוד היה יוצר את השדה המחושב עבור כל שורה בטבלה המקורית, אך לא היה מספק תוצאות סיכום לפי קבוצה (הוא היה מחשב את ההכנסה הכוללת של כל המוצרים, במקום את ההכנסה הכוללת של כל מוצר בנפרד).

הפסוקית ORDER BY שולטת בסדר המיון של קבוצת ההחזרה. למרות שניתן לכתוב קוד זה בצורה תמציתית (הקוד זהה לקוד שמחשב את מחיר הסיכום לשורה), קל יותר להבינו כפי שהוא כתוב, והוא דומה לקוד שיוצרת רשת עיצוב השאילתה.

פונקציות הגדרת נתונים

קיימות כמה דרכים להגדיר נתונים באמצעות משפטי SQL. בסעיף זה נבחן כיצד ליצור טבלה באמצעות שאילתות יצירת טבלה. שאילתות יצירת טבלה הן שאילתות פעולה שיוצרות טבלה חדשה המבוססת על קבוצת ההחזרה של שאילתה קיימת. זוהי פונקציית הגדרת נתונים.

בסעיף זה גם נדגים כיצד לשנות שדה שגודל בתוספת קבועה באופן אוטומטי באמצעות מילות המפתח של SQL – ALTER TABLE ו- ALTER COLUMN. Jet 4 SQL כולל סוג נתונים שדה זיהוי שמאפשר לעשות זאת. Jet 4 תומך בהגדרת ערכי צעד (step) והתחלה (start) עבור שדות מונה אלה (ניתן לאפס אותם בכל עת). שגרות הדוגמה בסעיף זה כוללות כבר את כל הרכיבים התפקודיים, אך ערכן יעלה אם תבחן אותן לאור האמור בפרק 3. ALTER TABLE ו- ALTER COLUMN הן חלק מקבוצה שלמה של מילות מפתח (לדוגמה, CREATE TABLE, CREATE INDEX ו-DROP) שתומכות ישירות בפונקציות הגדרת נתונים באמצעות SQL.

SELECT...INTO

לפניך התחביר של שאילתת יצירת טבלה באמצעות SQL :

```
SELECT FieldList INTO NewTableName FROM RecordSource
```

אם תיצור טבלה חדשה בקובץ מסד נתונים שונה מזה הנוכחי, הוסף פסוקית IN אחרי הפסוקית INTO ולפני הפסוקית FROM. השתמש בפסוקית IN כדי לייעד את הנתיב ואת שם הקובץ של מסד הנתונים שישמור את פלט המשפט SELECT...INTO.

RecordSource יכול להתבסס על טבלה אחת או יותר, או שאילתה אחת או יותר ועליו להכיל את כל השדות שברצונך לכלול בטבלה החדשה שאתה יוצר. המשפט SELECT...INTO מעתיק את עיצוב השדות שנבחרו ואת נתוניהם לטבלה החדשה. אין באפשרותך ליצור שדות חדשים כלשהם באמצעות המשפט SELECT...INTO. תוכל ליצור בקלות את התבנית הכללית של שאילתה זו על ידי יצירתה בתצוגת העיצוב ולאחר מכן לעבור לתצוגת SQL ולהעתיק את משפט SQL אל שיגרה שמשתמשת באובייקט ADO. אם תאמץ גישה זו, לא תצטרך לתכנן את משפט SQL.

השיגרה שלפניך משתמשת בגירסה פשוטה של משפט SELECT...INTO כדי לגבות את הטבלה FamilyMembers שבמסד הנתונים הנוכחי אל טבלה חדשה שנקראת FMBackup.

```
Sub MyMakeTable()
Dim cnn1 As ADODB.Connection

' Reference connection.
   Set cnn1 = CurrentProject.Connection

' Execute SQL for maketable query.
   cnn1.Execute "SELECT FamilyMembers.* INTO " & _
      "FMBackup FROM FamilyMembers"
' This procedure fails if FMBackup already exists.

End Sub
```

למרבה הצער, בדרך כלל כדי ליצור ולהפעיל משפט SELECT...INTO לא די בשני משפטי ADO הללו. לדוגמה, שיגרה זו תיכשל אם הטבלה FMBackup כבר קיימת. השיגרה יכולה להיכשל גם אם משתמש נוסף יפתח את אחת הטבלאות. קשיים אלה ואחרים מחייבים מנגנון ללכידת שגיאות. הדוגמה שלפניך ממחישה מנגנון אפשרי ללכידת שגיאות ביישום המשתמש בשיגרה בעלת משפט SELECT...INTO.

```
Sub MyMakeTable2()
On Error GoTo Make2Err:
Dim cnn1 As ADODB.Connection

' Reference connection.
   Set cnn1 = CurrentProject.Connection

' Test for unanticipated errors.
'    Err.Raise 1

' Execute SQL for make-table query.
   cnn1.Execute "SELECT FamilyMembers.* INTO " & _
      "FMBackup FROM FamilyMembers"

Make2Exit:
' Close the connection and set it equal to nothing and exit the sub.
   cnn1.Close
   Set cnn1 = Nothing
   Exit Sub
```

```
Make2Err:
' Trap for table already exists.
    If Err.Number = -2147217900 Then
        cnn1.Execute "Drop Table FMBackup"
        Resume
    Else
        MsgBox "The program generated an unanticipated " & _
            "error. Its number and description are " & _
            Err.Number & ": " & Err.Description, vbCritical, _
            "Programming Microsoft Access 2000"
    End If
Resume Make2Exit
End Sub
```

שיגרה זו מבצעת משימה זהה לזו של השיגרה המקורית, אך היא לא תיכשל גם אם הטבלה FMBackup כבר קיימת. המטפל בשגיאות מזהה שגיאה מסוג זה ומוחק את הטבלה הקיימת בטרם תבצע את הפקודה פעם נוספת. במקרה של שגיאה אחרת, היישום יציג את המידע החיוני בתיבת הודעה כך שהמשתמש יוכל להודיע למפתח היישום על הבעיה.

השיגרה כוללת רכיב נוסף ללכידת שגיאות: מייד לאחר הקמת החיבור, הקוד יכול לקרוא לשיטה Raise של האובייקט Err (הרכיב מסומן בתור הערה, אך דאגתי להסיר את הסימון כדי להעריך את ביצועי מנגנון לכידת השגיאות). שימוש זה באובייקט Err יוצר שגיאה מלאכותית שמסייעת לגלות כיצד יש לעצב את השיגרה כך שתגיב לשגיאות בלתי צפויות.

הדוגמה גם סוגרת את החיבור ומגדירה את האובייקט Connection בתור Nothing. שתי פעולות אלו דומות בייעודן ומשלימות זו את זו. בעת סגירת חיבור, המשאבים ששימשו לצורך החיבור הופכים זמינים לשאר המערכת. יחד עם זאת, האובייקט נשאר בזיכרון, ומאפייניו אינם משתנים. בשלב מאוחר יותר, תוכל לפתוח את אותו אובייקט Connection עם הגדרות המאפיינים הקודמות שלו או עם הגדרות שונות חדשות. עליך להגדיר אובייקט בתור Nothing כדי להסיר אותו מהזיכרון.

שימוש ב- ALTER TABLE וב- ALTER COLUMN כדי לאפס שדות Autoincrement

מפתחים רבים לסביבת Access יתלהבו מאפשרויות השליטה החדשות בסוגי נתונים Counter (מונה) המשמשים כשדות זיהוי של רשומות. בפרק 3 למדנו כבר כיצד לקבוע את ערכי ההתחלה והצעד (תוספת) של שדות מונה. מייד נלמד דרכים נוספות להגדלת רמת השליטה בשדות אלה. השיגרה שלפניך מגדירה את ערכי ההתחלה והצעד של סוג הנתונים Counter.

```
Sub ResetCounter(intStart, intStep)
Dim cnn1 As ADODB.Connection
Dim strSQL
' Reference connection and execute SQL for view.
   Set cnn1 = CurrentProject.Connection
' Create SQL string that references passed arguments.
   strSQL = "ALTER TABLE FamilyMemberNames " & _
      "ALTER COLUMN FamID Identity " & _
      "(" & intStart & "," & intStep & ")"
' Execute the SQL statement to update the counter.
   cnn1.Execute strSQL
End Sub
```

השיגרה ResetCounter מדגימה כיצד להפוך פקודת SQL לדינמית על ידי הוספת ארגומנטים למשפט SQL. תהליך זה דומה להעברת ערכים אל שאילתת פרמטר. השיגרה ResetCounter משנה את ערך השדה FamID בטבלה FamilyMemberNames וקולטת שני ארגומנטים. ארגומט אחד הוא ערך שדה המונה החדש של הרשומה הבאה שתתווסף לטבלה; הארגומנט השני הוא גודל הצעד עבור הרשומות הבאות החדשות. באפשרותך להפעיל את השיגרה על ידי הקלדת שורה מעין זו בחלון המיידי:

```
ResetCounter 2,2
```

שני ארגומנטים אלה מאלצים מתן ערך 2 לשדה FamID ברשומה הבאה שמתווספת לטבלה. כל ערך שדה FamID בכל רשומה נוספת יגדל ב-2 לעומת השדה ברשומה שקדמה לו. היזהר בקביעת ערכי ההתחלה והצעד, מכיון שאתה עלול לגרום להפרת כללי מפתח ראשי (כגון ערכים כפולים) וכך לשלול מהמשתמשים את הזכות להוסיף רשומות חדשות לטבלה. למרבה המזל, Access 2000 מאפשר לסקור באמצעות קוד את ערכי שדה המונה הקיימים, ולכן תוכל ליצור קוד שיהיה מוגן מפני בעיה זו. השיגרה הבאה מדגימה זאת.

משפט SQL הוא לב-ליבה של ResetCounter. כפי שניתן לראות, השיגרה כוללת שלושה מרכיבים קריטיים. מילות המפתח ALTER TABLE משנות את עיצוב הטבלה שצוינה. מילות המפתח ALTER COLUMN משנות את עיצוב השדה שצוין. מילת המפתח IDENTITY מעדכנת את שדה המונה בהתאם לארגומנטים של הערך ההתחלתי וערך הצעד של השדה.

השיגרה SetResetCounter שלפניך ארוכה, אך אינה מורכבת ביותר. היא מנצלת שיטה שמטפלת בערכי שדה מונה מבלי לגרום להפרות כללי מפתח רשומה. תרשים 4.26 מציג את הטבלה כפי שנראית מייד לאחר הפעלה מוצלחת של השיגרה SetResetCounter. שים לב כי השדה FamID ברשומה הראשונה מכיל את הערך 2, ואילו השדה FamID ברשומה הבאה אחריה מכיל ערך גדול בצעד של 2. השדה FamID ברשומה השלישית מכיל את הערך 8, ולכן אינו עקבי לערך הצעד. בנוסף, ערך השדה FamID ברשומות הבאות גדל בצעד של 4 בכל רשומה חדשה. השינוי בערכי ההתחלה והצעד של שדה המונה נובע מהקריאה ל-ResetCounter. מחרוזת טקסט בשדה MyMemoField מסמנת את השינוי בערכים אלה.

FamilyMemberNames : Table

FamID	PersonName	MyMemoField
2	Rick Dobson	start, step = 2
4	Virginia Dobson	
8	Glen Hill	see my new start & step
12	Tony Hill	
16	Shelly Hill	
(AutoNumber)		

Record: 1 of 5

תרשים 4.26: הטבלה FamilyMemberNames לאחר ביצוע השיגרה SetResetCounter

```
Sub SetResetCounter()
Dim cnn1 As New ADODB.Connection
Dim cmd1 As ADODB.Command
Dim rst1 As ADODB.Recordset, rst2 As New ADODB.Recordset
Dim int1 As Integer
' Reference connection.
   Set cnn1 = CurrentProject.Connection
' Clear FamilyMemberNames table and reset table counter.
   DoCmd.SetWarnings False
   DoCmd.OpenQuery "qdlAllFamilyMemberNames"
   DoCmd.SetWarnings True
   ResetCounter 2, 2
' Add a couple of records to the FamilyMemberNamestable from the
' FamilyMembers table with the initial start and step values.
   Set rst1 = New ADODB.Recordset
   rst1.Open "FamilyMemberNames", cnn1, adOpenKeyset, _
      adLockOptimistic, adCmdTable
   rst2.Open "FamilyMembers", cnn1, adOpenForwardOnly, _
      adLockReadOnly, adCmdTable
   For int1 = 1 To 2
      rst1.AddNew
         rst1(1) = rst2(1) & " " & rst2(2)
         If int1 = 1 Then
            rst1("MyMemoField") = "start, step = 2"
         End If
         rst2.MoveNext
      rst1.Update
   Next int1
   rst1.Close
```

```
' Define and execute command to select all FamID field values
' with highest FamID value first --> last counter value.
    Set cmd1 = New ADODB.Command
    With cmd1
        .ActiveConnection = cnn1
        .CommandText = "Select FamID from FamilyMemberNames " & _
            "Order By FamID Desc"
        .CommandType = adCmdText
        .Execute
    End With
' Save last counter value, and use it to reset start and step to new values.
    Set rst1 = New ADODB.Recordset
    rst1.CursorType = adOpenForwardOnly
    rst1.LockType = adLockReadOnly
    rst1.Open cmd1
    int1 = rst1(0)
    rst1.Close
    ResetCounter int1 + int1, int1

' Add remaining records from the FamilyMembers table
to the FamilyMemberNames
' table with the new start and step values.
    rst1.Open "FamilyMemberNames", cnn1, adOpenKeyset, _
        adLockOptimistic, adCmdTable
'    rst1(2) = "see my new start & step"
    int1 = 3
    Do Until rst2.EOF
        rst1.AddNew
            rst1(1) = rst2(1) & " " & rst2(2)
            If int1 = 3 Then
                rst1("MyMemoField") = "see my new start & step"
            End If
            rst2.MoveNext
            int1 = int1 + 1
        rst1.Update
    Loop
```

השיגרה SetResetCounter כוללת שישה חלקים המופרדים בהערות. החלק הראשון מקים חיבור אל מסד הנתונים הנוכחי. החלק השני מפעיל שאילתת מחיקה שמסירה את כל השורות מתוך הטבלה FamilyMemberNames, מאפשרת להציג אזהרות מערכת על ידי קריאה לשיטה SetWarning, ולאחר מכן מגדירה את ערכי ההתחלה והצעד של המונה של הטבלה FamilyMemberNames על ידי קריאה לשיגרה ResetCounter.

החלק השלישי משתמש בלולאה For...Next כדי להעתיק את שתי הרשומות הראשונות של הטבלה FamilyMembers אל הטבלה FamilyMemberNames. מנגנון מסד הנתונים Jet מקצה אוטומטית ערכי FamID לשתי הרשומות הללו באמצעות ערכי ההתחלה והצעד שהוגדרו בחלק הקודם של השיגרה.

החלק השלישי מפעיל את האובייקט Command כדי למצוא את השדה FamID בעל הערך הגדול ביותר בטבלה FamilyMemberNames. כשהפקודה מתבצעת, היא ממיינת את רשומות הטבלה בסדר יורד לפי השדה FamID. כתוצאה מכך, הרשומה בעלת הערך הגדול ביותר בשדה FamID – המפתח הראשי של הרשומה – מוצבת בראש קבוצת ההחזרה. ידיעת ערך זה מאפשרת ליישום לקבוע ערך התחלה חדש עבור רשומות נוספות, כזה שאינו משכפל את ערך המפתח הראשי של רשומה כלשהי בטבלה.

החלק החמישי פותח ערכת רשומות בקבוצת ההחזרה של הפקודה ומאחסן את ערך השדה FamID של הרשומה הראשונה בקבוצה. לאחר מכן, חלק זה מפעיל את השיגרה ResetCounter כדי להגדיר ערך התחלה חדש כפול מהערך הגדול ביותר בשדה FamID וגם ערך צעד חדש ששווה לערך הצעד הגדול ביותר בשדה FamID.

החלק האחרון מוסיף את הרשומות שנותרו בטבלה FamilyMembers לטבלה FamilyMemberNames. כדי להגדיל את גמישות השליטה בשדות מפתח ראשי מסוג מונה, העמודה Memo מכילה הודעה המציינת כיצד השיגרה SetRetCounter מטפלת ללא בעיה בשדה המונה.

תצוגות ושגרות מאוחסנות

Access 2000 כולל תצוגות ושגרות מאוחסנות שהיו פעם זמינות רק במערכות מתקדמות לניהול מסדי נתונים. Access 2000 מיישם תצוגות, שאינן תומכות בפרמטרים, בתור שאילתות מאוחסנות שמחזירות שורות. Access 2000 כולל שגרות מאוחסנות בתור שאילתות פעולה מאוחסנות וגם שאילתות פרמטר הפועלות כשאילתות פעולה או שאילתות בחירה. כפי שתיווכח לדעת, ניתן להשיג תפקודיות של שאילתת פרמטר על ידי העברת קבועי מחרוזת לשיגרה שממזגת את הערכים המועברים אל משפט SQL, וביצוע המשפט.

תצוגות

המשפט CREATE VIEW יוצר תצוגה על ידי הוספת שאילתת בחירה מאוחסנת לחלון מסד הנתונים תחת קטגוריית האובייקטים **שאילתות** (Queries) (בפרויקט Access מסוג adp.*, התצוגות שמורות תחת קטגוריית האובייקטים **תצוגות** (Views) של חלון מסד הנתונים). תהליך היצירה של שאילתת בחירה שמורה באמצעות CREATE VIEW ישיר יותר מאשר באמצעות ADO, מה שמחייב יצירת אובייקט מסוג Command והוספתו לאוסף **Views**.

השיגרה שלפניך מדגימה את התחביר ואופן הפעולה של משפטי CREATE VIEW בקוד SQL של ADO.

```
Sub CreateView()
Dim cnn1 As ADODB.Connection
' Reference connection.
   Set cnn1 = CurrentProject.Connection
' Execute SQL for view.
   cnn1.Execute "Create View CategoryView as SELECT * From Categories"
   RefreshDatabaseWindow

'This simple routine fails if CategoryView already exists.
End Sub
```

השיגרה מפעילה את השיטה Execute של החיבור ומעבירה אליה משפט CREATE VIEW. שם שאילתת הבחירה בא מייד לאחר מילות המפתח CREATE VIEW ואחריו מילת המפתח AS ומשפט SELECT. המשפט SELECT יכול להיות משפט SELECT כלשהו מהסוג המקובל. לבסוף, השיגרה קוראת ל-RefreshDatabaseWindow כדי לעדכן את חלון מסד הנתונים כדי שהמשתמש יוכל להציג את השאילתה החדשה מבלי שיצטרך לרענן מחדש את החלון.

ל-CreateView יש חיסרון: היא נכשלת אם התצוגה שהיא מנסה ליצור, קיימת כבר. באפשרותך, כמובן, להסיר בעצמך את השאילתה הישנה, אך ניתן למחוק אותה גם באמצעות קוד. שתי השגרות שלפניך ממחישות גישה אפשרית לפתרון הבעיה:

```
Sub CreateReplaceView(ViewName As String)
Dim cnn1 As ADODB.Connection
Dim cat1 As ADOX.Catalog
Dim vew1 As ADOX.View
' Reference objects for view
   Set cnn1 = CurrentProject.Connection
   Set cat1 = New Catalog
   cat1.ActiveConnection = cnn1

 Enumerate views.
   For Each vew1 In cat1.Views
' Delete named view and replace it with a new version.
      If vew1.Name = ViewName Then
         cat1.Views.Delete ViewName
         CreateCustomView ViewName
         Exit Sub
      End If
   Next vew1

' If the view is not there, create from scratch.
   CreateCustomView ViewName
   RefreshDatabaseWindow
End Sub
```

```
Sub CreateCustomView(MyViewName)
Dim cnn1 As ADODB.Connection, strSQL
' Reference connection and execute SQL for view.
   Set cnn1 = CurrentProject.Connection

' Create SQL string for view.
   strSQL = "Create View " & MyViewName & " as Select * From Categories"
' Create the custom view.
   cnn1.Execute strSQL
   RefreshDatabaseWindow

End Sub
```

השגרה CreateReplaceView אינה מבצעת ישירות את המשפט CREATE VIEW; היא משאירה משימה זו לשיגרה השנייה, CreateCustomView. CreateReplaceView ממספרת את התצוגות כדי להעריך את הצורך למחוק את התצוגה הישנה לפני שמנסים ליצור תצוגה חדשה ששמה זהה. לאחר מכן היא קוראת לשיגרה CreateCustomView ומעבירה את שם התצוגה שיש ליצור. CreateCustomView זהה בעיצובה לשיגרה שבדוגמה הקודמת, אך היא מנצלת שרשור מחרוזת עם הארגומנט המועבר וקבוע מחרוזת, כדי להרכיב משפט SQL עבור השיטה Execute של החיבור.

יוצרים תצוגה חדשה על ידי הקלדה של פקודה פשוטה בחלון Immediate, הנה כך:

```
CreateReplaceView "NameOfView"
```

שם התצוגה יופיע בין מרכאות כפולות. ניתן לשנות בקלות את CreateReplaceView כדי לאפשר משפט SELECT דינמי.

במקום לעבור בצורה מחזורית על מרכיבי האוסף Views, תוכל פשוט לנסות ליצור תצוגה חדשה וללכוד את השגיאה שנגרמת כתוצאה מתצוגה בעלת שם זהה. מספר השגיאה של תצוגה כפולה הוא 2147467259-. לכוד מספר שגיאה זה ומחק את התצוגה הישנה בטרם תבצע מחדש את המשפט CREATE VIEW. השיגרה FieldView בתקליטור המצורף לספר ממחישה גישה זו.

שגרות מאוחסנות

שיגרה מאוחסנת מפשטת את השימוש החוזר בקוד SQL. שגרות מאוחסנות מבצעות פעולות כלליות, כגון שאילתות מחיקה, עדכון או הוספה וקולטות פרמטרים בזמן ריצה הן עבור פעולות כלליות והן עבור שאילתות החזרת-שורות.

משתמשים במשפט CREATE PROC כדי ליצור שיגרה מאוחסנת. התחביר הכללי של משפט זה זהה לזה של המשפט CREATE VIEW. לאחר המילים CREATE PROC מקלידים את שם השיגרה המאוחסנת ואת מילת המפתח AS, המשמשת עבור סמן ההתחלה של קוד SQL המגדיר את התנהגות השיגרה המאוחסנת. באפשרותך לעצב שאילתות פעולה בתצוגת עיצוב השאילתה של Access, להעתיק את קוד SQL בתצוגת SQL ולאחר מכן להדביק את הקוד שהעתקת אחרי מילת המפתח AS.

שגרות מאוחסנות אינן מופיעות בחלון **מסד נתונים** (Database). לעומת זאת, הן קיימות בתור פריטים באוסף **Procedures** של האובייקט **Catalog**. הקוד יכול להתייחס לשגרות יחידות באמצעות סכמת אובייקט היררכית זו. לדוגמה, אם היישום יוצר שיגרה מאוחסנת שמוחקת רשומות, השיגרה קיימת באוסף **Procedures** (יחד עם שאילתות הפעולה בפרויקט).

השיגרה שלפניך מציגה את התחביר שמיישם את המשפט CREATE PROC. השיגרה מפעילה את השיטה Execute של האובייקט Connection ומעבירה משפט SQL בתור מחרוזת. המשפט CREATE PROC מנסה ליצור שאילתה שנקראת DeleteThese. השאילתה מוחקת את כל הרשומות של הטבלה FamilyMemberNames שערך השדה FamID שלהן הוא 10 או יותר. אם לא ננקוט אמצעים מיוחדים, השיגרה תיכשל במידה שהשאילתה DeleteThese כבר קיימת. כדי לאפשר להפעיל את השיגרה גם אם היא כבר קיימת, הקוד ממשיך להתבצע מהשורה הבאה במקרה של שגיאה.

```
Sub CreateProcToDelete()
On Error Resume Next
Dim cnn1 As ADODB.Connection
Dim cat1 As Catalog
Dim proc1 As Procedure
Dim cmd1 As Command
' Reference connection.
   Set cnn1 = CurrentProject.Connection
   Set cat1 = New Catalog
   cat1.ActiveConnection = cnn1
   Set cmd1 = New Command
   cmd1.ActiveConnection = cnn1

' Execute SQL to make the procedure.
   cnn1.Execute "Create Procedure DeleteThese As " & _
      "Delete From FamilyMemberNames Where FamID>=10"

' Enumerate Procedures collection members.
' For DeleteThese procedure, set command properties and execute it.
   For Each proc1 In cat1.Procedures
      If proc1.Name = "DeleteThese" Then
         cmd1.CommandText = proc1.Name
         cmd1.CommandType = adCmdStoredProc
         cmd1.Execute
      End If
   Next proc1
End Sub
```

השיגרה CreateProcToDelete מתחילה בסדרת הצהרות והצבות שתפקידן לתמוך בלוגיקה ליצירה והפעלה של השיגרה. השיטה Execute של האובייקט Connection יוצרת את השיגרה על ידי הפעלת משפט SQL. בשלב זה, אם השיגרה כבר קיימת, תתרחש שגיאה. השיטה Execute מוסיפה בפועל את השיגרה DeleteThese לאוסף **Procedures** בקטלוג של החיבור.

שגרות שוכנות באוסף, ולכן ניתן למספר את חברי האוסף באמצעות לולאת For...Each. הדוגמה משתמשת בלולאה כזו כדי לאתר את DeleteThese. ברגע שמוצאת את השיגרה, היא מציבה את המאפיין **Name** (שם) של השיגרה במאפיין CommandText של האובייקט Command. לאחר מכן היא מגדירה את המאפיין CommandType של האובייקט Command בתור adCmdStoredProc. שתי הצבות אלו יוצרות את מחרוזת SQL שנקראת Execute DeleteThese. הפעלת השיטה Execute של האובייקט Command מפעילה את השיגרה ב-Jet ומסירה את הרשומות מהטבלה FamilyMemberNames.

באפשרותך ליצור גירסה גמישה ונבונה של הדוגמה הקודמת בכמה דרכים. ראשית, מנגנון משוכלל יותר ללכידת שגיאות יכול לבודד סוגי שגיאות אחרים שמקורם בשיגרה. שנית, באפשרותך לגרום לשיגרה לקבל פרמטרים עבור השדה FamID. הדוגמאות שלפניך מנצלות את המשפט CREATE PROC כדי להגדיר שאילתת פרמטר שמוחקת את כל הרשומות שבטבלה FamilyMemberNames שערך השדה FamID שלהן שווה או גדול מערך שהוגדר בזמן ריצה. הקוד מזכיר את שלבי הגדרת ערך פרמטר של פקודה. שים לב כי עליך ליצור את הפרמטר, להוסיפו לאוסף **Parameters** ולאחר מכן להציב בו ערך. שגרת הדוגמה גם לוכדת שגיאות שמקורן בניסיון ליצור שיגרה ששמה זהה לשם של שיגרה קיימת.

```
Sub CreateProcToDelete2()
On Error GoTo Delete2Err
Dim cnn1 As ADODB.Connection
Dim cmd1 As ADODB.Command
Dim prm1 As Parameter

'Reference connection
   Set cnn1 = CurrentProject.Connection

'Test for unanticipated errors
'   Err.Raise 1

'Execute SQL to make the procedure
   cnn1.Execute "Create Proc DeleteThese " & _
      "(Parameter1 Long) As " & _
      "Delete From FamilyMemberNames " & _
      "Where FamID>=Parameter1"
```

```
'Assign SQL from procedure to command
    Set cmd1 = New ADODB.Command
    With cmd1
        .ActiveConnection = cnn1
        .CommandText = "DeleteThese"
        .CommandType = adCmdStoredProc
    End With

'Set the procedure's parameter
    Set prm1 = cmd1.CreateParameter("Parameter1", _
        adInteger)
    cmd1.Parameters.Append prm1
    prm1.Value = 10

'Invoke the procedure's SQL statement in the command
    cmd1.Execute

Delete2Exit:
    Exit Sub

Delete2Err:
    If Err.Number = -2147217900 Then
'Trap for procedure already exists
        cnn1.Execute "Drop Proc DeleteThese"
        Resume
    Else
        MsgBox "The program generated an unanticipated " & _
            "error.  Its number and description are " & _
            Err.Number & ": " & Err.Description, vbCritical, _
            "Programming Microsoft Access 2000"
    End If
    Resume Delete2Exit
End Sub
```

בעת ציון פרמטר במשפט CREATE PROC, הכלל את הצבת הפרמטר לאחר שם השיגרה, אך לפני מילת המפתח AS. באפשרותך להצהיר את שמה ואת סוג הנתונים שלה בפרמטרים, כפי שניתן לראות בדוגמה. מפנים שנית לפרמטר בפסוקית WHERE, כדי להגביל את פעולת השיגרה. התחביר זהה לזה של שאילתות פרמטר שיוצרים בתצוגת **עיצוב**.

שים לב כי אינך צריך בעצם לעבור בלולאה על האוסף **Procedures** כדי להפעיל שיגרה ספציפית. באפשרותך להתייחס לשמה כל עוד תגדיר את adCmdStoredProc בתור הערך של מאפיין הפקודה, CommandType. הדוגמה מציבה את הקבוע 10 בפרמטר (Parameter1) כדי לשמור על עקביות עם הדוגמה הקודמת, אך תוכל להשתמש במשפט הפונקציה InputBox או בטופס כדי לקלוט את הערך עבור הפרמטר.

הדוגמה גם כוללת מנגנון בסיסי ללכידת שגיאות למצב שקיימת כבר שיגרה בעלת שם זהה (מספר שגיאה 2147217900-). השיגרה CreateProcToDelete2 עושה דבר פשוט: היא משליכה את השיגרה הישנה במקרה של שגיאה על ידי הפעלת המשפט DROP PROC. תחביר המשפט הוא "DROP PROC *ProcedureName*".

הפעלת שאילתות על מקורות נתונים מרוחקים

עד כה, הדוגמאות שהצגנו בפרק התייחסו למקורות נתונים מקומיים, או כאלה שנמצאים בקובץ Access אחר. Access 2000 מאפשר לעבוד עם גם מקורות נתונים מרוחקים, כגון Microsoft SQL Server או Oracle. בסעיף זה נעסוק בנושא בצורה תמציתית; בפרק 12 נתייחס אליו בהרחבה.

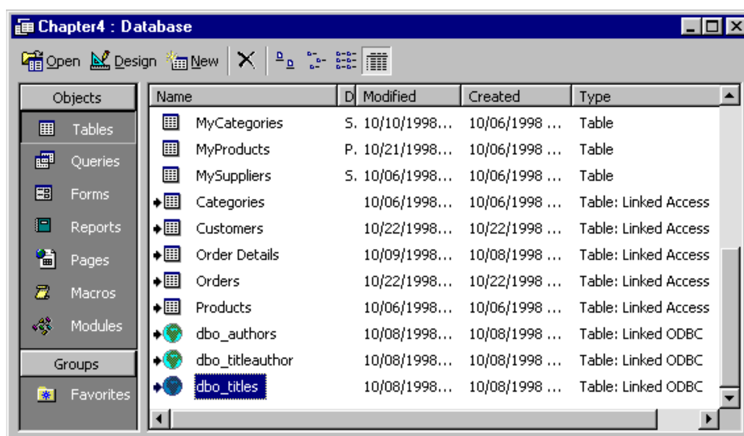
הפעלת שאילתות על מקורות ODBC מקושרים

באפשרותך לבצע שאילתה על מקור ODBC מקושר בדיוק כפי שאתה עושה עם מקור נתונים של Access שנמצא בקובץ אחר. תרשים 4.27 מציג את חלון מסד הנתונים ובו שלוש טבלאות מקושרות ממסד נתונים Pubs שמצורף למוצר SQL Server. פעל בהתאם לשלבים שלהלן כדי לקשר את היישום לטבלה שנמצאת במסד נתונים מרוחק:

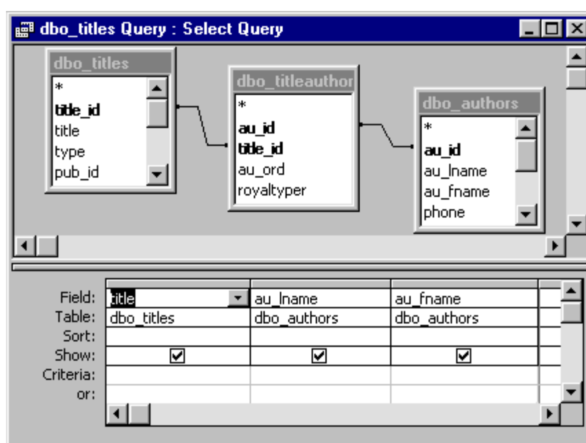
1. בחר באפשרות **קבל נתונים חיצוניים** (Get External Data) מתפריט **קובץ** (File) של Access ולאחר מכן לחץ על **קישור טבלאות** (Link Tables).
2. בתיבת הדו-שיח **קישור** (Link), בחר באפשרות **ODBC Databases** מהרשימה הנפתחת **קבצים מסוג** (Files Of Type).
3. בחר ב-DSN בכרטיסיה **Machine Data Source** (באפשרותך ליצור DSN חדש אם זה הדרוש לך אינו זמין).
4. השלם את הקישור על ידי בחירת טבלה ולחיצה על **OK**.

מאחר שכמעט כל מקורות הנתונים תומכים ב-ODBC, דרך טיפול זו בנתונים מרוחקים מספקת גישה אוניברסלית באמצעות הממשק המוכר של Access.

החלון העליון בתרשים 4.28 מציג את שלוש הטבלאות המקושרות שבתרשים 4.27 בתצוגת עיצוב שאילתה. לאחר יצירת הקישור אל טבלה מרוחקת, באפשרותך להתייחס אליה כאילו היתה טבלה מקומית. טבלאות במקורות נתונים מרוחקים עלולות להיות גדולות מאוד, ולכן יש להתחשב בהן בצורה מיוחדת בעת עיצוב שאילתות המפנות אליהן. הסעיף הראשון בפרק זה שופך אור על כמה טכניקות אפשריות. גודל הטבלאות וסוג החיבור הפיסי למקור הנתונים המרוחק יסייע לקבוע את עיצוב השאילתה הטוב ביותר. החלון התחתון בתרשים 4.28 מציג את השאילתה שבחלון העליון בתצוגת **גיליון נתונים**.



תרשים 4.27: חלון מסד נתונים זה מציג שלוש טבלאות מקושרות – dbo_authors, dbo_titleauthor ו-dbo_titles. באפשרותך להשתמש במנהלי התקנים של ODBC כדי ליצור קישור אל טבלאות שנמצאות במקור נתונים כלשהו באמצעות מנהל התקן של ODBC



dbo_titles Query : Select Query

title	au_lname	au_fname
The Busy Executive's Database Guide	Bennet	Abraham
Fifty Years in Buckingham Palace Kitchens	Blotchet-Halls	Reginald
But Is It User Friendly?	Carson	Cheryl
The Gourmet Microwave	DeFrance	Michel
Silicon Valley Gastronomic Treats	del Castillo	Innes
Secrets of Silicon Valley	Dull	Ann
The Busy Executive's Database Guide	Green	Marjorie
You Can Combat Computer Stress!	Green	Marjorie

Record: 1 of 25

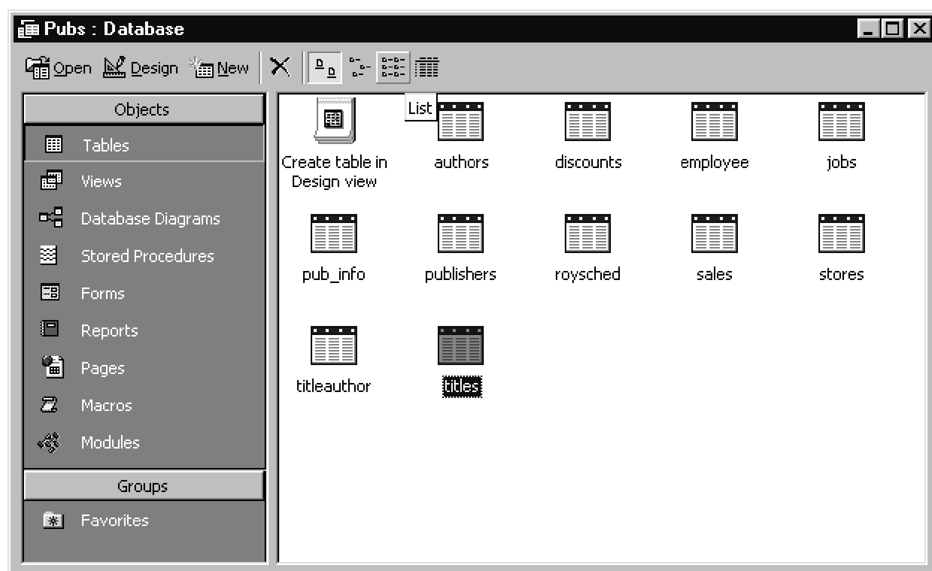
תרשים 4.28: שאילתה פשוטה המבוססת על שלוש טבלאות ממקור נתונים מרוחק. החלון העליון מראה כי ניתן ליצור שאילתות אל טבלאות מקושרות מרוחקות כאילו היו מקומיות. החלון התחתון מציג את השאילתה בתצוגת **גיליון נתונים**

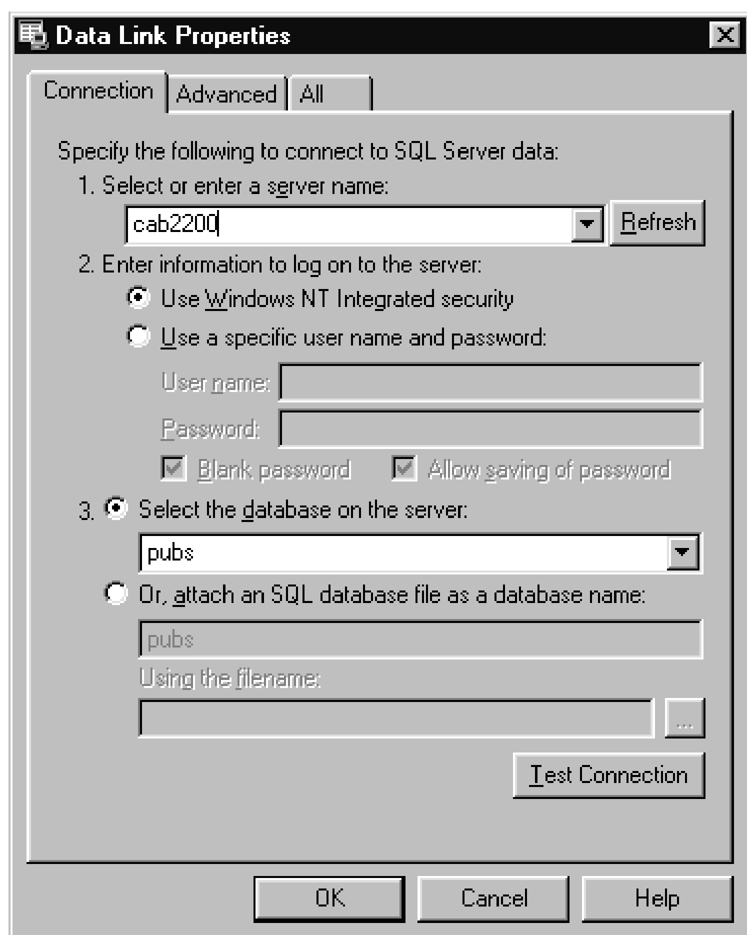
הפעלת שאילתות בפרויקט נתונים של Access

Access 2000 כולל סוג פרויקט חדש שנקרא Access Project, אשר עוקף את מנגנון מסד הנתונים Jet על ידי יצירת קישור ישיר אל גרסאות 6.5 או 7.0 של SQL Server וגם אל Microsoft Data Engine. Microsoft Data Engine הוא הגירסה החדשה של SQL Server עבור סביבת Windows 95/98. סביבה זו מאפשרת ליצור קישור ישיר אל מסד נתונים מרוחק ולטפל בו כאילו היה מסד נתונים מקומי.

החלון העליון בתרשים 4.29 מציג את חלון **מסד נתונים** של פרויקט Access. שים לב שהוא זהה במראהו לחלון **מסד נתונים** של מסד נתונים ‎*.mdb, אך מקור הנתונים שלו הוא מסד הנתונים Pubs (באפשרותך להחליף את מסד הנתונים שיהיה מקושר אל חלון מסד הנתונים על ידי בחירת **התקשרות** (Connection) מתפריט **קובץ** (File)).

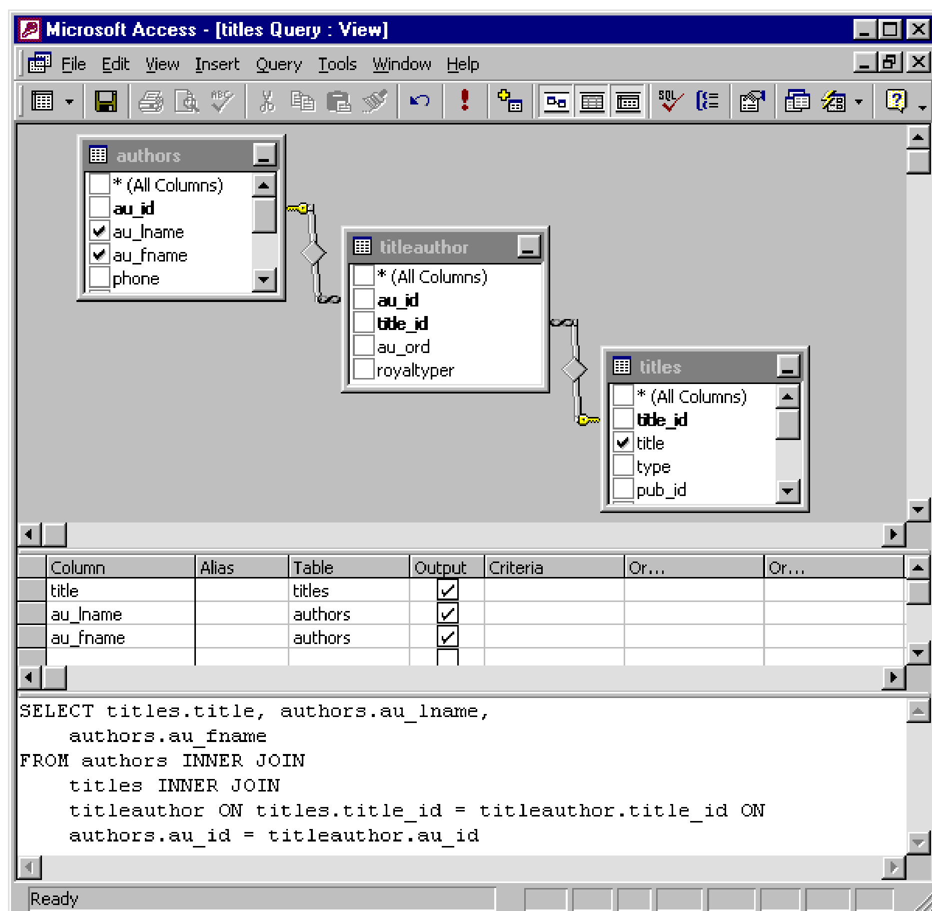
החלון התחתון בתרשים 4.29 הוא תיבת הדו-שיח **Data Link Properties** של הפרויקט. השרת של פרויקט זה הוא cab2200, שם שרת מסד הנתונים SQL Server 7.0 המקושר. כדי להקים חיבור פעיל אל שרת, עליך לציין את מידע הכניסה (logon). הדוגמה הנוכחית מנצלת את האבטחה המשולבת של Windows NT. לאחר שסיימת להקים את החיבור, תוכל לפתוח מסד נתונים כלשהו שנמצא בשרת שמותר לך להיכנס אליו. לחץ על **Test Connection** כדי לוודא את חוקיות הקישור למסד הנתונים המבוקש.





תרשים 4.29: החלון הראשון הוא חלון **מסד נתונים** של Access Project; החלון השני הוא תיבת הדו-שיח **Data Link Properties** של הפרויקט

תרשים 4.30 מציג את תצוגת העיצוב של שאילתה ב- Access Project. רשת עיצוב השאילתה שלו דומה לזו של מסד נתונים מסוג mdb.*, אך היא מהווה חלק מאוסף כלי העיצוב החזותיים, Microsoft Da Vinci. באפשרותך ללחוץ לחיצה ימנית כדי להוסיף טבלאות או שאילתות נוספות בתור מקורות רשומות לעיצוב של שאילתה חדשה, ותוכל גם לגרור ולשחרר שדות בין טבלאות כדי ליצור קשרי גומלין. לא קיימת תצוגת SQL מפורשת לעניין זה, אך ניתן להחליף את מצב ההבחנה של חלונית SQL בתצוגת העיצוב באמצעות לחצן בסרגל הכלים **מסד נתונים** (Database). הלחצן הראשון בסרגל כלים זה עדיין מאפשר להחליף בין התצוגה **עיצוב** (Design) והתצוגה **גיליון נתונים** (Datasheet). כיוון רשת עיצוב השאילתה משתנה מאנכי לאופקי, אך העיצוב נשאר דומה. בונים תצוגות על ידי הוספת קבועים וביטויים לתאי הקריטריונים מימין לשמות השדות ברשת עיצוב השאילתה.



תרשים 4.30: תצוגת **עיצוב** של שאילתה ב- Access Project. שים לב להצגת קוד SQL של שאילתה בחלונית נפרדת ולא בתצוגה נפרדת

תכנות שאילתות עבור מסדי נתונים מרוחקים

מפתחים רבים מעדיפים לעבוד ישירות עם ממשק תכנות ולא עם ממשק משתמש גרפי. Access 2000 מציע תאימות הדוקה יותר בין שאילתות של מקורות נתונים מקומיים ומרוחקים ברמת הקוד לעומת זו שברמת ממשק המשתמש הגרפי, מכיון שניתן להשתמש ב-ADO וב-SQL בעבודה עם שני סוגי מקורות הנתונים. ההבדל הגדול ביותר בין גישה לנתונים מקומיים או מרוחקים, הוא שגישה לנתונים מרוחקים מחייבת לציין את פרמטרי OLE DB של ספק OLE DB (OLE DB provider) בציריך הנתונים שברצונך לתשאל. העברת יישומים ממקור נתונים מרוחק עלולה להצריך טיפול בתאימות שפת SQL, כיון שמנגנוני מסדי נתונים מרוחקים שונים דורשים לפעמים פרמטרי כניסה (logon) שונים ומשתמשים בהרחבות SQL ייחודיות עבור הרכיבים המיוחדים שלהם.

כדי להתחבר למקור נתונים מסוג Microsoft SQL Server, תזדקק לספק OLE DB מיוחד שנקרא SQLOLEDB, אשר מותאם במיוחד לעבודה עם Microsoft SQL Server, במקום מנהל ההתקן הכללי של ODBC, MSDASQL, המיועד לעבודה עם מסדי נתונים ODBC. מנהל ההתקן MSDASQL ותיק יותר מ-SQLOLEDB, אך הוא עדיין מנהל התקן רב תכליתי למצבים בהם לא קיים מנהל התקן OLE DB ספציפי. כל אחד משני מנהלי ההתקנים הללו מקבל פרמטרים שונים במקצת (בפרק 3 הסברנו כיצד לקבוע את הפרמטרים עבור MSDASQL).

הדוגמה שלפניך ממחישה את הישירות של תכנות שאילתה למקור נתונים מרוחק מסוג SQL Server 7. שים לב שהשיגרה משתמשת במנהל ההתקן SQLOLEDB. השאילתה מבוצעת מול מסד הנתונים Pubs בשרת cab2200. לאחר פתיחת החיבור בהתבסס על מנהל ההתקן SQLOLEDB והגדרות הפרמטרים שלו, השיגרה יוצרת אובייקט מסוג Command ומגדירה את המאפיינים CommandText ו-CommandType שלו. לאחר מכן היא מבצעת את הפקודה, מציבה את קבוצת ההחזרה באובייקט Recordset ומדפיסה את קבוצת ההחזרה.

```
Sub SQLOleDBQuery()
Dim cnn1 As ADODB.Connection
Dim cmd1 As ADODB.Command
Dim rst1 As ADODB.Recordset
' Establish a connection to the database.
' Specify the server (cab2200) as the data source and catalog as the database.
   Set cnn1 = New ADODB.Connection
   With cnn1
      .Provider = "sqloledb"
      .ConnectionString = "data source=cab2200;" & _
         "user id = sa;initial catalog=pubs"
      .Open
   End With
' Set up a command object.
   Set cmd1 = New ADODB.Command
   With cmd1
      .ActiveConnection = cnn1
      .CommandText = "SELECT titles.title, " & _
         "authors.au_lname, authors.au_fname " & _
         "FROM titles INNER JOIN (authors INNER JOIN " & _
         "titleauthor ON authors.au_id = titleauthor.au_id) ON " & _
         "titles.title_id = titleauthor.title_id"
      .CommandType = adCmdText
   End With
' Open and print a recordset based on the executed query statement.
   Set rst1 = cmd1.Execute
   Debug.Print rst1.GetString
End Sub
```

שאילתה זו מפיקה תוצאות זהות לאלו של השאילתות המוצגות בתרשימים 4.28 ו-4.30. לקחתי את קוד SQL שבדוגמה הנוכחית ישירות מתצוגת SQL של השאילתה שבתרשים 4.28. שים לב שהיא דומה, אך אינה זהה בארגון שלה, לקוד SQL שבתרשים 4.30 של Access Project. יחד עם זאת, באפשרותך להמיר את קוד SQL מתרשים 4.30 לתוך השיגרה שלעיל מבלי לשנות את קבוצת ההחזרה. הדבר מאשר את הדעה שגישת התכנות המבוססת על mdb.* וגישת Access Project יכולות להפיק תוצאות שקולות למרות כמה הבדלי סגנון בייצוג שלהן ב-SQL במקרים מסוימים.

5

בניית ממשק משתמש באמצעות טפסים

יישומים משתמשים בטפסים ובדוחות כדי לייצג נתונים. טפסים מהווים גם אמצעי לקליטה ומתן משוב על קלט משתמש. משתמשי מסד הנתונים מתקשרים עם היישום בעיקר באמצעות טפסים, ולכן יש חשיבות רבה לעיצוב טפסים ולאופן תפקודם. פרק זה פותח בהסבר כיצד ליצור רושם ראשוני מלהיב על ידי שימוש בטופס שנקרא **מסך פתיחה** (Splash Screen). בהמשך הפרק נלמד כיצד לגרום לטפסים (במיוחד טפסי איגוד-נתונים) לתקשר עם נתונים. נלמד גם על עיצוב מותנה וביצוע פעולות חיפוש מידע עם טפסים. לבסוף, נלמד להפנות לטפסים באמצעות קוד, לכבות ולהפעיל לסירוגין את המצב הגלוי שלהם וגם כיצד לעבוד עם מודולי מחלקת טופס.

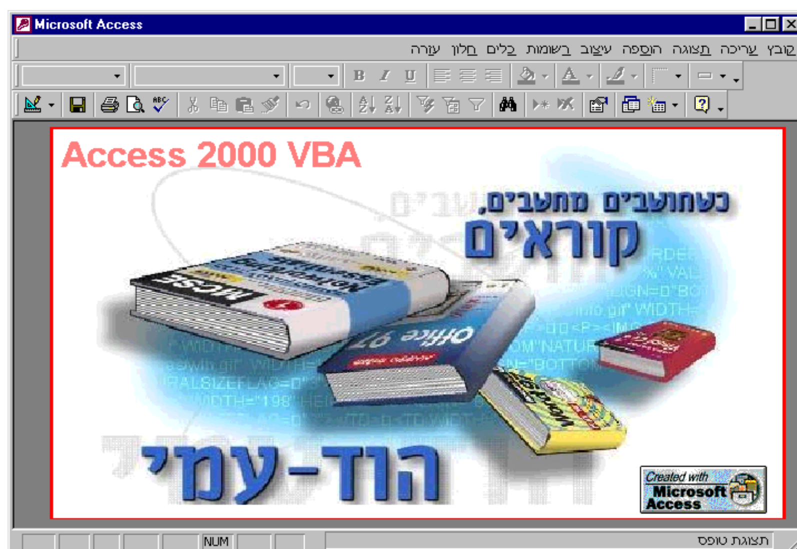


הערה:

Microsoft Access 2000 מציע שני סוגי טפסים, UserForms הזמינים ביישומים נוספים של Office, וטופס מותאם שהוא גירסה עדכנית של טופס מגרסאות מוקדמות של Access. UserForms אינם משתלבים בסביבת הפיתוח של Access בצורה כה הדוקה כמו טפסי Access. לדוגמה, UserForms אינם מאוגדים לטבלאות ולשאילתות Access. בנוסף, אשפי הטפסים ואשפי הפקדים של Access אינם מטפלים ב-UserForms. מסיבות אלו הפרק עוסק בטפסי Access בלבד. תוכל להרחיב את ידיעותיך על UserForms באמצעות העזרה המקוונת.

טופס מסך פתיחה

אחת הדרכים הקלות להתחיל לעבוד בצורה בסיסית עם טפסים היא על ידי יצירת מסך פתיחה. מסך פתיחה הוא טופס שמופיע טרם הצגת טופס אחר אינטראקטיבי יותר. מסכי פתיחה מתארים את ייעוד היישום ומציגים בדרך כלל גם את שם מחברו. תוכל לקבוע בקלות רבה את משך ההצגה של מסך פתיחה. תרשים 5.1 מציג את מסך הפתיחה שנמצא בקובץ Chapter5.mdb בתיקיה Code\Chap05\.. שבתקליטור המצורף. תוכל לאמץ אותו לצרכיך.



תרשים 5.1: מסך פתיחה לדוגמה

יצירת מסך פתיחה

תוכל להתחיל ביצירת מסך הפתיחה על ידי יצירת רקע פרוש. לשם כך עליך להגדיר שני מאפיינים בזמן שהטופס פתוח בתצוגת עיצוב. ראשית, הגדר למאפיין **תמונה** (Picture) (בכרטיסיה **תבנית** (Format) של גיליון המאפיינים) את הנתיב ושם קובץ התמונה. קובץ זה יכול להיות מפת סיביות בתבנית ‎*.bmp, *.ico, *.dib, *.wmf‎ או ‎*.emf‎. Access יכול גם להשתמש בתבניות קובץ אחרות שעבורם התקנת את המסננים הגרפיים הדרושים (תוכל להתקין את המסננים החסרים באמצעות תוכנית ההתקנה). תמונות המכילות גווני אפור או צבעים מעוממים אחרים מתאימות ביותר כתמונות רקע, מכיון שהן מבליטות את הרכיבים שבקידמתן – תמונות וטקסט. שנית, תן למאפיין הטופס **ריצוף תמונה** (Picture Tiling) את הערך **כן** (ערך ברירת המחדל שלו הוא **לא**) אם ברצונך שהתמונה תחזור על עצמה (לא בוצע בתרשים 5.1).

בשלב הבא, תוכל להוסיף תמונה בקידמה על ידי בחירה באפשרות **תמונה** (Picture) מתפריט **הוספה** (Insert). תוכל גם להוסיף פקד **ציור** (Image) לטופס. אם תנהג כך או

אחרת, תוצג תיבת הדו-שיח **הוספת תמונה** (Insert Picture) שממנה תוכל לבחור תמונה להוספה. לחיצה על **אישור** מגדירה אוטומטית את המאפיין **תמונה**. Access מאפשר להגדיר מאפיין זה באמצעות קוד VBA המאפשר לבנות את מראה הטופס בצורה דינמית בתגובה לקלט המשתמש (כגון קלט תיבת טקסט), או גורמים סביבתיים (כגון קוד האבטחה של המשתמש). תוכל למקם את התמונה, לשנות את גודלה ולהפעיל את האפקטים המיוחדים באמצעות הגדרות המאפיין **מצב שינוי גודל** (Size Mode). המאפיין יכול לקבל שלושה ערכים: **גזור** (Clip), **מתח** (Stretch) **ושנה מרחק מתצוגה** (Zoom).

נשלים את מסך הפתיחה על ידי הוספת פקד **תווית** (Label) אחד או יותר. תוכל להשתמש ב-VBA כדי להגדיר מאפיינים עבור פקדי **תווית** בזמן ריצה. הדבר מאפשר לעצב את טקסט מסך הפתיחה בצורה דינמית.

תוכל להגדיר את מסך הפתיחה כך שייפתח אוטומטית על ידי בחירת האפשרות **הפעלה** (Startup) מהתפריט **כלים** (Tools) ובחירה בשם הטופס של מסך הפתיחה מתוך הרשימה הנפתחת **הצג טופס/עמוד** (Display Form/Page) שבתיבת הדו-שיח **הפעלה**. באפשרותך גם להסתיר את חלון מסד הנתונים בעת פתיחת יישום, על ידי ביטול בחירת תיבת הסימון **הצג חלון מסד נתונים** (Display Database Window) בתיבת הדו-שיח **הפעלה**. לחץ על **אישור** כדי לשמור את הבחירות.

בקרה על משך התצוגה

זוג שגרות האירוע שלפניך מציג מסך פתיחה במשך 10 שניות (כדי להציג את קוד VBA שעליו מבוסס הטופס, לחץ לחיצה ימנית על הטופס, בחר באפשרות **אירוע בנייה** (Build Event) מתפריט הקיצור, לאחר מכן בחר באפשרות **בונה קוד** (Code Builder) בתיבת הדו-שיח **בחירת בונה** (Choose Builder) ולחץ על **אישור**). שגרת האירוע Form_Open נותנת למאפיין הטופס, TimerInterval, את הערך 10000 (שווה ל-10 שניות; מרווח הזמן מבוטא ביחידות מידה של אלפית השנייה). שגרת האירוע Form_Open סוגרת את הטופס. שים לב כי שגרת האירוע Form_Open משתמשת במילת המפתח Me כדי לציין את הטופס. באפשרותך להחליף את "frmSplashTimer" בערך Me.Name בשיגרה Form_Timer; מוסכמת השם Me עמידה יותר, מכיון שהיא מאפשרת לשנות את שם הטופס מבלי לשנות את הקוד.

```
Private Sub Form_Open(Cancel As Integer)
   Me.TimerInterval = 10000
End Sub


Private Sub Form_Timer()
   DoCmd.Close acForm, "frmSplashTimer"
End Sub
```

טופס מסך ניווט

טופסי מסך ניווט הם דרך מקובלת לאפשר ניווט ביישום. טפסים אלה יכולים להכיל לחצני פקודה אחדים שהמשתמש יכול ללחוץ עליהם כדי לפתוח טופס נוסף. בסעיף זה נציג שתי גישות לשימוש בטופסי מסך ניווט: האחת בעזרת היפר-קישורים והשנייה מבוססת על שגרות VBA.

ניווט בעזרת היפר-קישורים

קל מאוד ליצור ניווט באמצעות היפר-קישורים, מכיון שהדבר איו מחייב כתיבת קוד כלשהו (למרות שניתן לנהל היפר-קישור באמצעות VBA). היפר-קישורים יכולים לתפקד בתור קיצורי דרך אל אובייקטים של מסד נתונים ביישום, מסמכים בכונן הקשיח, קבצים ברשת או דפי Web באינטרנט או דפים ברשת האינטרא-נט הארגונית. Access מאפשר להשתמש בהיפר-קישורים עם תוויות, לחצנים ותמונות.

באפשרותך להגדיר ולערוך מאפייני היפר-קישור מתוך תצוגת העיצוב של טופס, או באמצעות קוד VBA. קל יותר להגדיר ולערוך היפר-קישורים באמצעות שגרות ידניות (המאפשרות גם להפיק טפסים הנטענים במהירות רבה יותר), מכיון שטפסים בעלי היפר-קישורים שנוצרו בתצוגת עיצוב אינם מחייבים שמודול יכיל קוד VBA. כדי לנצל יתרון זה, צריך מאפיין הטופס **יש מודול** (Has Module) להיות מוגדר בתור **לא**.

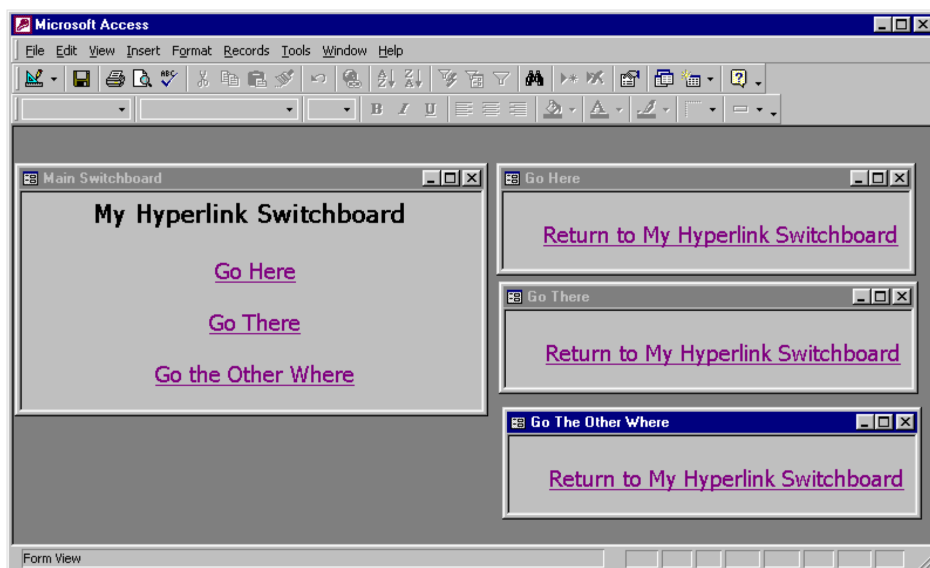


הערה:

סוג הנתונים **היפר-קישור** מאפשר ליישום Access להפעיל היפר-קישור מתוך שדות טבלה או שאילתה. שדות מסוג **היפר-קישור** דומים במובנים רבים למאפייני היפר-קישור, אך משתמשים בהם בצורה שונה. עיין בפרק 13 לקבלת מידע נוסף אודות שימוש בהיפר-קישורים לפיתוח בסביבת Web.

תרשים 5.2 מציג ארבעה טפסים המשתמשים במערכת ניווט פשוטה. טופס מסך הניווט הראשי שבצד שמאל מעביר את המוקד אל אחד משלושת הטפסים האחרים כשהמשתמש לוחץ על היפר-קישור בטופס הראשי. ברגע שהמוקד עובר לטופס אחר, יכול המשתמש להחזיר את המוקד אל מסך הניווט על ידי לחיצה על ההיפר-קישור המפנה אל הטופס הראשי.

באפשרותך להגדיל את מספר הרמות של היררכיית מערכת הניווט, בצורה קלה ופשוטה. באופן טיפוסי, כל מסך בן יכול להחזיר את המיקוד למסך האם שלו. מסכים ראשיים כוללים לעיתים קרובות דרך יציאה מהיישום או מ-Access.



תרשים 5.2: מערכת ניווט פשוטה בעזרת היפר-קישורים

כדי ליצור היפר-קישור באמצעות פקד **תווית** (Label), פעל כך :

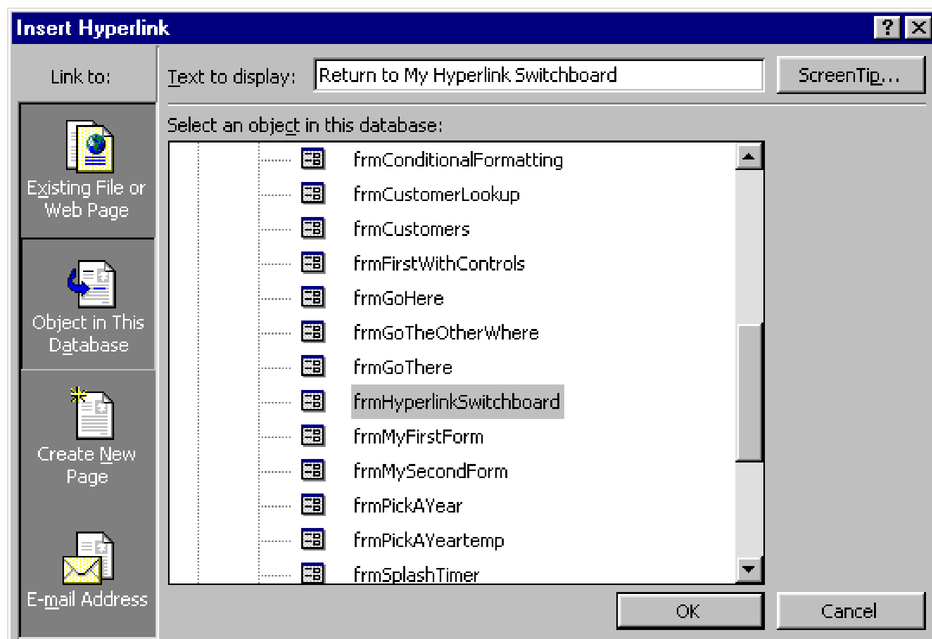
1. לחץ על לחצן **בנייה** (Build) שליד המאפיין **כתובת_היפר-קישור** (Hyperlink Address) או ליד המאפיין **כתובת משנה של היפר-קישור** (Hyperlink SubAddress) בגיליון המאפיינים של התווית.
2. בתיבת הדו-שיח **הוספת היפר-קישור** (Insert Hyperlink), בחר את סוג האובייקט לקישור מתוך הרשימה **קישור אל** (Link To).
3. הקלד את המידע עבור סוג האובייקט.
4. לחץ על **אישור**.



הערה:

כברירת מחדל, היפר-קישורים שלא לחצו עליהם בהפעלת Access מוצגים בכחול, וכאלה שלחצו עליהם – בצבע המכונה סיגלית. באפשרותך לשנות צבעים אלה על ידי בחירה ב**אפשרויות** (Options) מתפריט **כלים** (Tools), בחירה בכרטיסיה **כללי** (General) ולחיצה על **אפשרויות אינטרנט** (Web Options). השתמש בשתי תיבות הרשימה שבתיבת הדו-שיח **אפשרויות אינטרנט** כדי לקבוע את הצבעים הרצויים.

תרשים 5.3 מציג את תיבת הדו-שיח **הוספת היפר-קישור** בטופס frmGoHere במסד הנתונים המדגמי של פרק זה. ההיפר-קישור מעביר את המיקוד בחזרה אל טופס מסך הניווט.

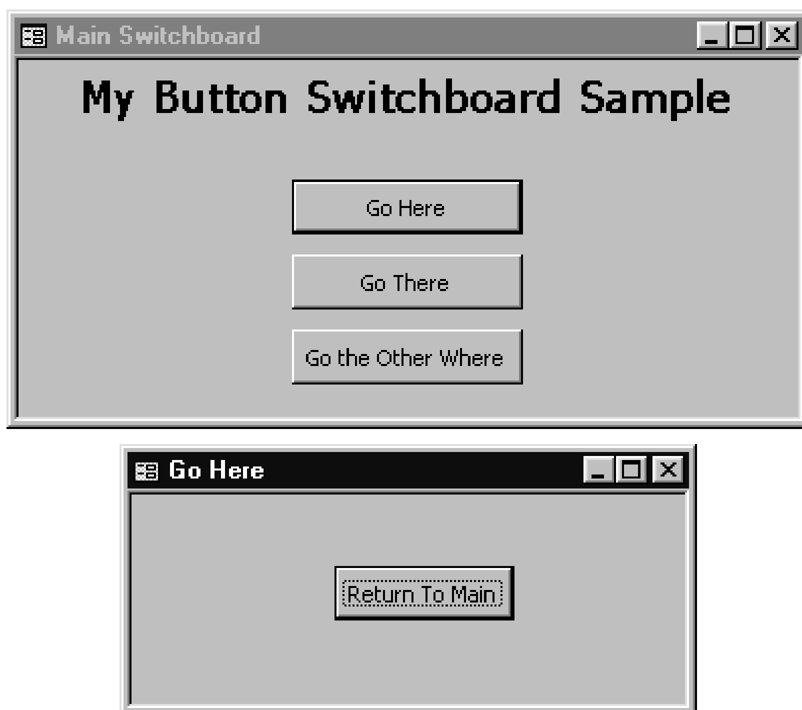


תרשים 5.3: תיבת הדו-שיח **הוספת היפר-קישור** עבור התווית בטופס frmGoHere

ניווט באמצעות קוד

דרך מקובלת נוספת למימוש ניווט באמצעות טופס מסך היא באמצעות קוד VBA מאחורי אירועי לחיצה של לחצן פקודה. גישה זו מאפשרת חשיפה עשירה יותר לרכיבים התפקודיים של Access לעומת ניווט שמבוסס על היפר-קישורים, מכיון שכך ניתן לשלב פונקציות ניווט עם אירועים נוספים, כגון סגירת טופס (היפר-קישור מאפשר פתיחת טופס מתוך טופס אחר, אך החזרה אל מסך הניווט מותירה את הטופס הבן פתוח; ניווט מבוסס-VBA מאפשר ליישום לסגור את טופס הבן כשהמשתמש סוגר את הטופס הראשי). שגרת אירוע מאפשרת גם להגדיל את השליטה באובייקטים רבים של מסד נתונים. ניווט המבוסס על היפר-קישורים פשוט מעביר את המיקוד לאובייקט אחר, כגון טופס.

תרשים 5.4 מציג זוג טפסים המממשים ניווט בגישת VBA. דוגמה זו מבוססת על אירועי לחיצה לחצן. כשהמשתמש לוחץ על אחד מלחצני מסך הניווט, קוד אירוע לחיצת הלחצן פותח את טופס היעד וסוגר את מסך הניווט הראשי. לחיצה על לחצן **Return To Main** בטופס היעד סוגרת את הטופס ופותחת את טופס הניווט הראשי (הדוגמה מבוססת על לחצני פקודה, אך באפשרותך להשתמש בבקרה מסוג אחר כלשהו שמאפשרת למשתמש ליצור אירועים).



תרשים 5.4: טפסים אלה משתמשים בקוד מאחורי אירועי לחיצת הלחצנים, כדי לנווט

זוג שגרות האירוע שלפניך מציג את קוד אירועי הלחיצה של שני הלחצנים שבתרשים 5.4. השיגרה הראשונה מעבירה את השליטה ממסך הניווט הראשי אל המסך השני. היא גם סוגרת את הטופס הראשי. השיגרה השנייה מעבירה את השליטה חזרה אל מסך הניווט הראשי ולאחר מכן סוגרת את הטופס השני. השתמשתי באשף לחצני הפקודה כדי ליצור את הגירסה הראשונית של כל אחת מהשגרות; לאחר מכן הוספתי שורת קוד לכל שיגרה כדי לסגור את המסך המתאים.

```
Private Sub cmdGoHere_Click()
On Error GoTo Err_cmdGoHere_Click
Dim stDocName As String
    stDocName = "frmButtonGoHere"
    DoCmd.OpenForm stDocName
    DoCmd.Close acForm, "frmButtonSwitchboard"
Exit_cmdGoHere_Click:
    Exit Sub
Err_cmdGoHere_Click:
    MsgBox Err.Description
    Resume Exit_cmdGoHere_Click
End Sub
```

```
Private Sub cmdReturnToMain_Click()
On Error GoTo Err_cmdReturnToMain_Click
Dim stDocName As String
Dim stLinkCriteria As String
   stDocName = "frmButtonSwitchboard"
   DoCmd.OpenForm stDocName
   stDocName = "frmButtonGoHere"
   DoCmd.Close acForm, stDocName, acSaveNo
Exit_cmdReturnToMain_Click:
    Exit Sub
Err_cmdReturnToMain_Click:
    MsgBox Err.Description
    Resume Exit_cmdReturnToMain_Click
End Sub
```

קישור טפסים לנתונים

מאז ומתמיד איפשר Access לאגד טפסים עם נתונים בצורה קלה ופשוטה. זו אחת הסיבות העיקריות להיותו סביבת פיתוח מהירה.

שימוש באשף הטפסים האוטומטיים

כדי לאגד טופס עם נתונים, ניתן להפעיל את **אשף הטפסים האוטומטיים** (AutoForm Wizard). בחר טבלה או שאילתה בחלון מסד הנתונים ולחץ על הלחצן **אובייקט חדש: טופס אוטומטי** (New Object:AutoForm) בסרגל הכלים **מסד נתונים** (Database). האשף יפתח טופס חדש שמאוגד ישירות למקור הנתונים שנבחר. תרשים 5.5 מציג טופס דוגמה המבוסס על הטבלה Order Details במסד הנתונים המדגמי שבפרק זה. באפשרותך לנצל טופס זה לעיון, עריכה, הוספה או מחיקה של רשומות בטבלה Order Details.

Order Details
Order ID 10248
Product Queso Cabrales
Unit Price $14.00
Quantity 12
Discount 0%
Record: 1 of 2155

תרשים 5.5: טופס שנוצר באמצעות אשף הטפסים האוטומטיים

מקור הנתונים של השדה Product בטופס הוא שדה בדיקת מידע בטבלת הרקע Order Details, ולכן השדה Product מוצג אוטומטית בתור תיבה משולבת המציגה את שמות המוצרים, במקום את ערכי הרקע ProductID. כל הרכיבים התפקודיים הללו סופקו אוטומטית על ידי האשף.

עיצוב מותנה

Access 2000 מאפשר לעצב בצורה מותנית את הנתונים המוצגים באמצעות פקד תיבת טקסט או תיבה משולבת ללא תכנות. באפשרותך להחיל עיצוב מותנה על פקדי טופס, בשדות מאוגדים או בלתי מאוגדים כאחד.

תרשים 5.6 מציג שלושה מופעים של השדות Discount ו- Extended Price. השדה Discount בטופס העליון הפך ללא זמין. הטופס האמצעי מבליט את הערך בשדה Extended Price באמצעות עיצוב של הטיה והדגשה. הטופס התחתון הופך את השדה Discount לזמין ומבליט את הערך שבשדה Extended Price.

השדה Extended Price מחושב; ערכו אינו נגזר ישירות מהטבלה שברקע. המאפיין **מקור פקד** (Control Source) של תיבת הטקסט מכיל ביטוי ([UnitPrice]*[Quantity]*(1-[Discount])) באמצעותו הוא מחשב את הערך בעת שהמשתמש עובר לרשומה חדשה, או מעדכן את השדות UnitPrice, Quantity או Discount ברשומה הנוכחית (המונחים בסוגריים המרובעים מתייחסים לפקדים ואינם שמות של שדות עבור מקור הנתונים שברקע).

frmConditionalFormatting : Form

Conditional Formatting Sample Form

Extended Price:

Order ID: 10248 $168.00

Product: Queso Cabrales

Unit Price: $14.00

Quantity: 12

Discount: 0%

Record: 1 of 2155

frmConditionalFormatting : Form

Conditional Formatting Sample Form

Extended Price:

Order ID: 10249 $1,696.00

Product: Manjimup Dried Apple

Unit Price: $42.40

Quantity: 40

Discount: 0%

Record: 5 of 2155

frmConditionalFormatting : Form

Conditional Formatting Sample Form

Extended Price:

Order ID: 10250 $1,261.40

Product: Manjimup Dried Apple

Unit Price: $42.40

Quantity: 35

Discount: 15%

Record: 7 of 2155

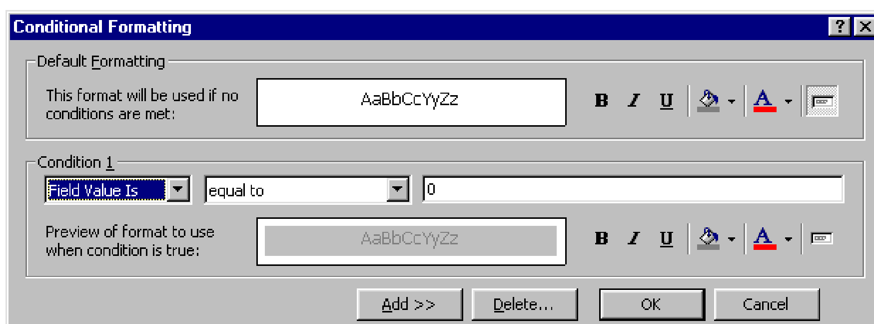
תרשים 5.6: עיצוב מותנה שולט במראה השדות Extended Price ו-Discount בטופס שלפניך



הערה:

מפתחים לא מנוסים נותנים לעיתים שמות זהים לשדות ולפקדים. הרגל זה עלול לבלבל ולגרום לשגיאות (אשף הטפסים האוטומטיים לוקה אף הוא בחיסרון זה). מומלץ להוסיף תחיליות לשמות פקדים כדי להבדילם משמות השדות ברקע שלהם. לדוגמה, שמו של פקד תיבת טקסט שמאוגד עם שדה ששמו UnitPrice יהיה txtUnitPrice.

כדי להחיל עיצוב מותנה על פקד, בחר את הפקד ובחר באפשרות **עיצוב מותנה** (Conditional Formatting) מתפריט **עיצוב** (Design), כדי לפתוח את תיבת הדו-שיח **עיצוב מותנה**, המוצגת בתרשים 5.7. לכל פקד בעל עיצוב מותנה יש שני עיצובים לפחות – עיצוב ברירת מחדל ועיצוב מיוחד שחל כאשר מתקיים תנאי מוגדר. באפשרותך לעצב את הפקד בהתאם לערך השדה של הפקד, לערך הביטוי או לעיתוי קבלת המיקוד. בעת טיפול בערך השדה של פקד, באפשרותך לבחור מתוך רשימת אופרטורים של השוואה, כגון שווה ל (=), גדול מ (>) וקטן מ (<). התנאי של השדה Discount שבתרשים 5.7 הוא **ערך שדה הינו שווה ל 0** (Field value is equal to 0). אפשרות העיצוב שנבחרה לתנאי זה מבטלת את זמינות הפקד כאשר ערך ההנחה (discount) הוא 0.



תרשים 5.7: תיבת הדו-שיח **עיצוב מותנה**

אם תחיל עיצוב מותנה על שדה מחושב, כגון Extended Price, עליך לכתוב ביטוי באמצעות אופרטורים תקניים של VBA. התנאי עבור השדה Extended Price הוא (Expression is text4.value>500) Text4 הוא הפקד שמציג את הערך המחושב). כשערך השדה גדול מ-500, התוכן שבתיבת הטקסט יובלט בעזרת הדגשה והטיה.

באפשרותך להחיל על פקד תנאי נוסף ועיצוב מיוחד. כל שעליך לעשות לשם כך הוא ללחוץ על לחצן **הוסף** (Add) בתיבת הדו-שיח **עיצוב מותנה** (Conditional Formatting) ולציין את התנאי החדש ואת נתוני העיצוב שלו.

טפסי משנה

טופס משנה (subform), המספק את אחת הדרכים המקובלות ביותר להצגת נתונים ב-Access, הוא טופס המוטבע בטופס ראשי. הטופס הראשי מכיל מידע כללי אודות אובייקט (כגון הזמנה או שם של מטופל). פרט היררכי קשור אחד או יותר (כגון שורות פריט בהזמנה או ביקורי המטופל) מופיע בטופס משנה אחד או יותר של הטופס הראשי. שדה משותף אחד לפחות חייב לקשור יחדיו את מקור הרשומות של הטופס הראשי ולכל אחד מטפסי המשנה. השדה המשותף מאפשר לטופס המשנה להציג רק את הרשומות שתואמות לרשומה הנוכחית בטופס הראשי. כשהמשתמש עובר לרשומה חדשה בטופס הראשי, טופס המשנה מציג קבוצת רשומות חדשה שקשורה בצורה ייחודית לרשומה החדשה בטופס הראשי.

תרשים 5.8 מציג מסך ראשי, MyOrders, המכיל טופס משנה מוטבע ששמו MyOrderDetails. הטופס MyOrders מקשר את השאילתה MyOrders אל השאילתה MyOrderDetails בהתאם לשדה המשותף OrderIDField (שאילתות אלו וטבלאות הרקע שלהן לקוחות ממסד הנתונים Northwind). בעת יצירת הטופס הראשי וטופס המשנה, לא יצרתי קשרי גומלין בין שתי השאילתות בחלון **קשרי גומלין** (Relationships) או על ידי שימוש בגליונות נתונים משניים (עיין בפרק 4 לקבלת מידע נוסף על גליונות נתונים משניים).

תרשים 5.8: טופס המכיל טופס משנה

כדי ליצור טופס משנה, פתח את הטופס הראשי בתצוגת עיצוב, ודא שהלחצן **אשפי בקרה** (Control Wizards) שב**ארגז הכלים** (Toolbox) לחוץ, ולאחר מכן גרור טבלה, שאילתה או טופס מתוך חלון מסד הנתונים ושחרר אותו בטופס הראשי. טופס המשנה יוצג בתור פקד בטופס הראשי. כדי לסנכרן את הטופס הראשי עם טופס המשנה, עליך להקצות שדה משותף אחד לפחות. בחר את מכולת טופס המשנה והגדר את המאפיינים **קישור שדות צאצא** (Link Child) ו**קישור שדות מראשי** (Link Master) בתור השדה המשותף. בשאילתות MyOrders ו-MyOrderDetails, השדה המשותף הוא OrderID.

טופס ראשי יכול להכיל טפסי משנה רבים. הדרישה היחידה היא שמקור הרשומות של כל טופס משנה יחלוק שדה משותף אחד לפחות עם מקור הרשומות של הטופס הראשי. לדוגמה, אם המסך הראשי שבתרשים 5.8 היה מכיל את השדה EmployeeID, הטופס היה יכול לכלול טופס משנה נוסף המבוסס על הטבלה Employees.

אם אתה מגדיר קשרי גומלין בין טבלאות ושאילתות בחלון קשרי הגומלין, או על ידי שימוש במאפייני גיליון משני, באפשרותך ליצור מסך ראשי עם טופס משנה מוטבע באותה קלות בה תיצור טופס מאוגד פשוט. בחלון מסד נתונים, בחר את הטבלה או את השאילתה שעליה יתבסס הטופס הראשי, ולאחר מכן לחץ על הלחצן **אובייקט חדש: טופס אוטומטי** (New Object:AutoForm). אשף הטפסים האוטומטיים יבנה טופס ראשי ובו טופס משנה מוטבע. טופס המשנה מנצל את המידע שבחלונות קשרי הגומלין או בגיליון הנתונים המשני. באפשרותך לגרור טבלאות, שאילתות או טפסים נוספים באופן ידני אל הטופס הראשי בתצוגת העיצוב כדי ליצור טפסי משנה נוספים.

בדיקת מידע והצגת נתונים

טופס בדיקת מידע דומה לשאילתת פרמטר בעלת ממשק קדמי מותאם אישית; הטופס אוסף קלט שמפעיל שאילתה. בעת שימוש בטפסים וב-VBA, באפשרותך לפעול בצורה גמישה בכל הנוגע לאופן איסוף המידע וגם באשר לסוג המידע שתוכל להחזיר למשתמשי היישום.

יצירת טופס בדיקת מידע

הדרך הקלה ביותר למימוש טופס שמבצע בדיקת מידע היא לתת למשתמש להקליד את פרטי בדיקת המידע בתיבת טקסט וללחוץ על לחצן כדי להתחיל בחיפוש. תיבת הטקסט חייבת להיות בלתי מאוגדת, מכיון שהיא אינה מכניסה מידע למסד הנתונים; כל שהיא עושה הוא לקבל נתוני חיפוש מהמשתמש. המידע שבתיבת הטקסט ינוצל על ידי שאילתה למציאת המידע המבוקש.

תרשים 5.9 מציג טופס שפותח את השאילתה qprHistoryfromTextBox כשהמשתמש מקליד קוד זיהוי בתיבת הטקסט ולוחץ על הלחצן **Look it up**. השאילתה מוצאת את הכמות הכוללת שהוזמנה מכל מוצר שקנה הלקוח.

frmTextBoxLookup : Form

Enter a customer ID and click Look It Up to find the total quantity ordered of each product bought by the customer

Customer ID: bergs

Look it up

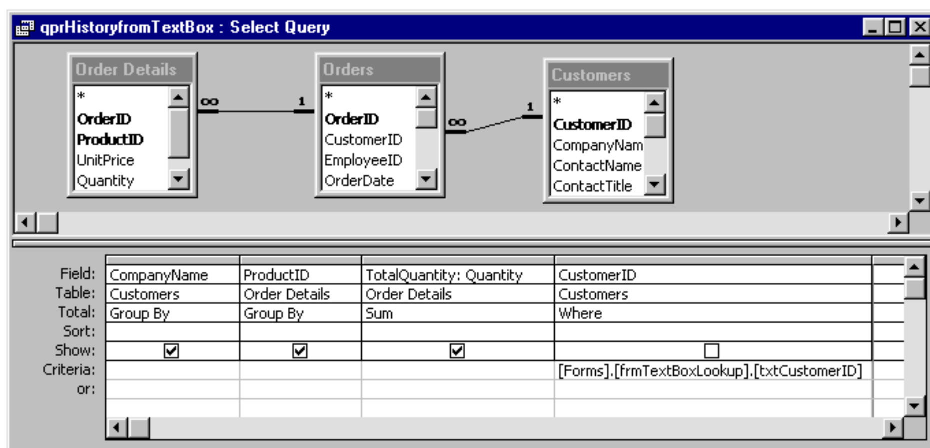
qprHistoryfromTextBox : Select Query

Company Name	Product	TotalQuantity
Berglunds snabbköp	Chai	35
Berglunds snabbköp	Chang	21
Berglunds snabbköp	Aniseed Syrup	30
Berglunds snabbköp	Chef Anton's Cajun Seasoning	12
Berglunds snabbköp	Ikura	22
Berglunds snabbköp	Konbu	6
Berglunds snabbköp	Tofu	16
Berglunds snabbköp	Pavlova	12
Berglunds snabbköp	Alice Mutton	10

Record: 1 of 37

תרשים 5.9: טופס שמבצע בדיקת מידע פשוטה

תרשים 5.10 מציג את השאילתה qprHistoryfromTextBox בתצוגת עיצוב. שים לב כי מדובר בשאילתה פשוטה שמסכמת את השדה Quantity בטבלה Order Details עבור כל אחד מהמוצרים שהלקוח רכש (הפסוקית WHERE בעמודה האחרונה מגבילה את השאילתה לשורות שהשדה CustomerID שלהן מתאים לערך שבתיבת הטקסט של הטופס).



תרשים 5.10: השאילתה qprHistoryfromTextBox בתצוגת עיצוב

האלמנט האחרון בטופס הוא שגרת אירוע VBA המופעלת כשהמשתמש לוחץ על לחצן Look It Up. השיגרה כוללת שורה יחידה הפותחת את השאילתה qprHistoryfromTextBox:

```
Private Sub cmdLookup_Click()
   DoCmd.OpenQuery "qprHistoryfromTextBox"
End Sub
```

שימוש בתיבה משולבת עבור קלט משתמש

הטופס שבתרשים 5.11 מעוצב טוב יותר מזה שתרשים 5.9. במקום לאלץ את המשתמש לנחש את קוד זיהוי הלקוחות, הוא מאפשר לו לבחור את הלקוח מתוך תיבה משולבת. שגרת האירוע After Update של התיבה המשולבת פותחת את השאילתה qprHistoryfromTextBox שמשתמשת בנתוני הלקוח שבחר המשתמש. לכן, לחצן הפקודה אינו נחוץ יותר.

frmComboBoxLookup : Form

Select a customer to find the total quantity ordered of each product bought by the customer

Company Name:

Alfreds Futterkiste
Ana Trujillo Emparedados y helados
Antonio Moreno Taquería
Around the Horn
Berglunds snabbköp
Blauer See Delikatessen
Blondel père et fils
Bólido Comidas preparadas

תרשים 5.11: עיצוב משופר של טופס בדיקת מידע



הערה:

טפסי בדיקת מידע המכילים תיבה משולבת עלולים לפגוע בביצועים ככל שעולה מספר פריטי בדיקת המידע. במצב זה עומדות לרשותך שתי אפשרויות. הראשונה, לחזור אל טופס בדיקת המידע בעל תיבת הטקסט שתואר לעיל. לחילופין, תוכל ליצור מערכת מרובדת שתאפשר למשתמשים לבחור אפשרויות שיגבילו את טווח הפריטים הזמינים בתיבה המשולבת.

כדי להעניק תפקודיות זו לפקד תיבה משולבת, פעל כך :

1. הגדר את המאפיין **סוג מקור שורה** (Row Source Type) של הפקד בתור **טבלה/שאילתה** (Table/Query) (ברירת המחדל).
2. הגדר את המאפיין **מקור שורה** (Row Source) של הפקד בתור מחרוזת SQL שמחזירה את השדות המבוקשים (מחרוזת SQL של הדוגמה היא SELECT CUSTOMERID FROM CUSTOMERS).

3. תן למאפיין **מונה עמודות** (Column Count) של הפקד את הערך 2.

4. הגדר את רוחבי העמודות והפרד אותן בפסיקים (הערך הראשון חייב להיות 0).

אם תעדיף, אשף התיבות המשולבות יכול ליצור את הפקד עבורך. כל שעליך לעשות הוא לוודא שלחצן **אשפי בקרה** (Control Wizards) שבארגז הכלים (Toolbox), במצב לחוץ, ולאחר מכן הוסף את התיבה המשולבת לטופס. עקוב אחר הוראות האשף.

הצגת תוצאות בתיבת הודעה

הדוגמאות שהיצגנו לוקות בשתי חולשות. ראשית, הן מעבירות ערכים לשאילתות וגם חושפות ערכים הלקוחים משאילתות. משמעות הדבר, שמשתמשים עלולים לשבש בשוגג את עיצוב השאילתה. שנית, משתמש יכול גם לשנות את הנתונים שברקע השאילתה.

הטופס המדגמי לבדיקת מידע שבחלקו העליון של תרשים 5.12 מתגבר על שני החסרונות הללו על ידי שימוש ב-VBA וב- ActiveX Data Objects (ADO). טופס הקלט דומה במראהו ובתפקודו לטופס שבתרשים 5.11. קבוצות ההחזרה של השאילתות שפותחים שני הטפסים זהות, אך הן מוצגות בדרכים שונות. הדוגמה שבתרשים 5.12 מציגה את קבוצת ההחזרה בתיבות הודעה במקום בחלון שאילתה בתצוגת **גיליון נתונים** (הדוגמה עושה שימוש במספר תיבות הודעה הדרוש כדי להציג את קבוצת ההחזרה שלה). באופן זה נתוני הרקע מוגנים מפני שיבוש שלא במתכוון על ידי המשתמש.

frmComboLookupSQL : Form

Select a customer to find the total quantity ordered of each product bought by the customer

Company Name: Berglunds snabbköp





תרשים 5.12: הטופס הראשון מציג את קבוצת התוצאות שלו בתיבות הודעה כך שהמשתמשים יכולים להציג את קבוצת התוצאות, אך לא לשנות נתוני הרקע שלה

השיגרה שלפניך מופעלת בתגובה לאירוע After Update של התיבה המשולבת בתרשים 5.12. השיגרה מבצעת פקודה בהתבסס על שאילתה, כדי לפתח קבוצת החזרה. לאחר מכן היא מקצה את קבוצת ההחזרה מהפקודה אל ערכת רשומות ומציגה את ערכת הרשומות באמצעות תיבת הודעה אחת או יותר.

```
Private Sub cboLookup_AfterUpdate()

Dim ctl1 As Control
Dim cmd1 As Command
Dim rst1 As Recordset, str1 As String

' Set reference to ComboBox control.
Set ctl1 = Me.Controls("cboLookup")

' Create and define command.
' Use ComboBox value in SQL string for command.
    Set cmd1 = New ADODB.Command

    With cmd1
        .ActiveConnection = CurrentProject.Connection
        .CommandText = "Select Customers.CompanyName, " & _
            "Products.ProductName, " & _
            "Sum([Order Details].Quantity) As TotalQuantity " & _
            "From Products Inner Join ((Customers Inner Join Orders " & _
            "ON Customers.CustomerID = Orders.CustomerID) " & _
            "Inner Join [Order Details] ON " & _
            "Orders.OrderID = [Order Details].OrderID) " & _
            "ON Products.ProductID = [Order Details].ProductID " & _
            "Where Customers.CustomerID = '" & ctl1.Value & "'" & _
            "GROUP BY Customers.CompanyName, Products.ProductName;"
        .CommandType = adCmdText
        .Execute
    End With

' Create recordset based on return set from SQL string.
    Set rst1 = New ADODB.Recordset
    rst1.Open cmd1, , adOpenKeyset, adLockOptimistic

' Loop through return set to display in message box(es)
' in blocks of 925 characters or less.
    Do Until rst1.EOF
        str1 = str1 & rst1.Fields(0) & ", " & rst1.Fields(1) & ", " & rst1.Fields(2)
        str1 = str1 & vbCrLf
```

```
      If Len(str1) > 925 Then
         str1 = str1 & vbCrLf & "Click OK to see more in another message box"
         MsgBox str1, vbInformation, "Programming Microsoft Access 2000"
         str1 = ""
      End If

      rst1.MoveNext
   Loop

   MsgBox str1, vbInformation, _
      "Programming Microsoft Access 2000"
End Sub
```

בשיגרה זו, לא יכולתי לנצל את קוד SQL מחלון שאילתה בתצוגת **עיצוב**, מכיון שמחרוזת SQL של אובייקט מסוג Command אינה תומכת בשדות בדיקת מידע. לכן, הוספתי את הטבלה Products לעיצוב השאילתה כך שיכולתי לדווח על כל שם של מוצר בקבוצת ההחזרה במקום על קוד המוצר בלבד, כפי שהגיע מהטבלה Order Details. הטבלה הנוספת מסבכת את לוגיקת הצירוף של השאילתה (עיין בפרק 4 במבוא לתחביר של משפטי SQL).

לולאת Do עוברת בצורה סדרתית על ערכת הרשומות וכותבת את תכניה במחרוזת. בסוף כל רשומה, היא מכניסה תו החזרת גררה (CR) והזנת שורה (LF). אם אורך המחרוזת עולה על 925 תווים, השיגרה מכניסה שורה ריקה והוראה להצגת המשך היסטוריית המכירות של הלקוח בקטע ההודעה הבא. תיבת הודעה יכולה להכיל קצת יותר מ-1,000 תווים (השיגרה testmsgbox במסד הנתונים המדגמי בפרק זה מסייעת לקבוע את מספר התווים המקסימלי שתיבת הודעה יכולה להכיל; כל אחת מ-19 השורות העליונות כוללת שני תווים שאינם מודפסים). הגבלת הוספת תווים לתיבת ההודעה הנוכחית ל-925 תווים, מאפשרת לתיבה להתמלא מבלי לקצץ תו כלשהו.

הצגת מידע דינמית

באפשרותך להציג נתונים, כגון רשומה, בטופס, ואף לעצב טופס שיאפשר למשתמשים להציג את הרשומה, אך לא לערוך אותה. תרשים 5.13 מציג זוג טפסים שמאפשרים יחד למשתמש לעיין ברשומת לקוח. המשתמש בוחר לקוח מתוך הטופס frmCustomerLookup ולוחץ על הלחצן Show Customer In Form כדי לפתוח את הטופס Customers, אשר מציג את רשומת הלקוח שנבחר (המאפיינים **אפשר עריכה** (Allow Edits), **אפשר מחיקות** (Allow Deletions) **ואפשר תוספות** (Allow Additions) מקבלים את הערך **לא**, ולכן המשתמש מנוע מלשנות את הנתונים). המשתמש יכול ללחוץ על הלחצן Return To Customer Lookup Form כדי להעביר את השליטה חזרה אל טופס בדיקת המידע ההתחלתי. באפשרותו גם להפעיל בדיקת מידע נוספת, או לצאת מהיישום דרך הטופס הנוכחי.

frmCustomerLookup : Form

Select a customer in the combo box
and click Show Customer In Form to open a form that
displays the record for the selected customer

CompanyName: Blondel père et fils

Show Customer in Form

Close Customer Lookup Form

Customers

Read-only Customer Form

Customer ID	BLONP
Company Name	Blondel père et fils
Contact Name	Frédérique Citeaux
Contact Title	Marketing Manager
Address	24, place Kléber
City	Strasbourg
Region	
Postal Code	67000
Country	France
Phone	88.60.15.31
Fax	88.60.15.32

Return to Customer Lookup Form

תרשים 5.13: טפסים אלה מאפשרים למשתמש לבחור ולהציג רשומה של הלקוח

שגרת האירוע הפשוטה והאלגנטית שלפניך היא הקוד שמאחורי הלחצן Show Customer in Form.

```
Private Sub cmdShowCustomer_Click()
On Error GoTo ShowCustomerTrap
Dim strValue As String, strMsg As String
   strValue = Me.Combo2.Value
   DoCmd.OpenForm "frmCustomers", _
   acNormal, ,"CustomerID = '" & strValue & "'"
ShowCustomerTrapExit:
   Exit Sub
```

```
ShowCustomerTrap:
    If Err.Number = 94 Then
        MsgBox "Select a customer in the combo box " & _
            "before attempting to open the Customer form.", _
            vbExclamation, "Programming Microsoft Access 2000"
    Else
        strMsg = "Error number: " & Err.Number & "caused " & _
            "failure. Its description is:" & vbCrLf & Err.Description
        MsgBox strMsg, vbExclamation, "Programming Microsoft Access 2000"
    End If
    Resume ShowCustomerTrapExit

End Sub
```

יצירת תרשים מקבוצות משנה של נתונים

האובייקט Chart של Microsoft Graph 2000 מאפשר ליצור תרשימים בעלי מראה מקצועי, בצורה קלה ופשוטה. האובייקט, המשולב בטופס כפקד אובייקט בלתי מאוגד, יכול להיות מאוגד לטבלאות ושאילתות של Access. באפשרותך לבחור מתוך מיגוון רחב של סוגי תרשימים ואפשרויות עיצוב (לחץ לחיצה כפולה על האובייקט בטופס בתצוגת העיצוב כדי לחשוף את התפריט המותאם אישית של האובייקט. כשתסיים להשתמש בתפריט האובייקט Chart, לחץ על הטופס מחוץ לאובייקט כדי לשחזר את תפריט עיצוב הטפסים הרגיל של Access).

יצירת תרשים באמצעות אשף הטפסים

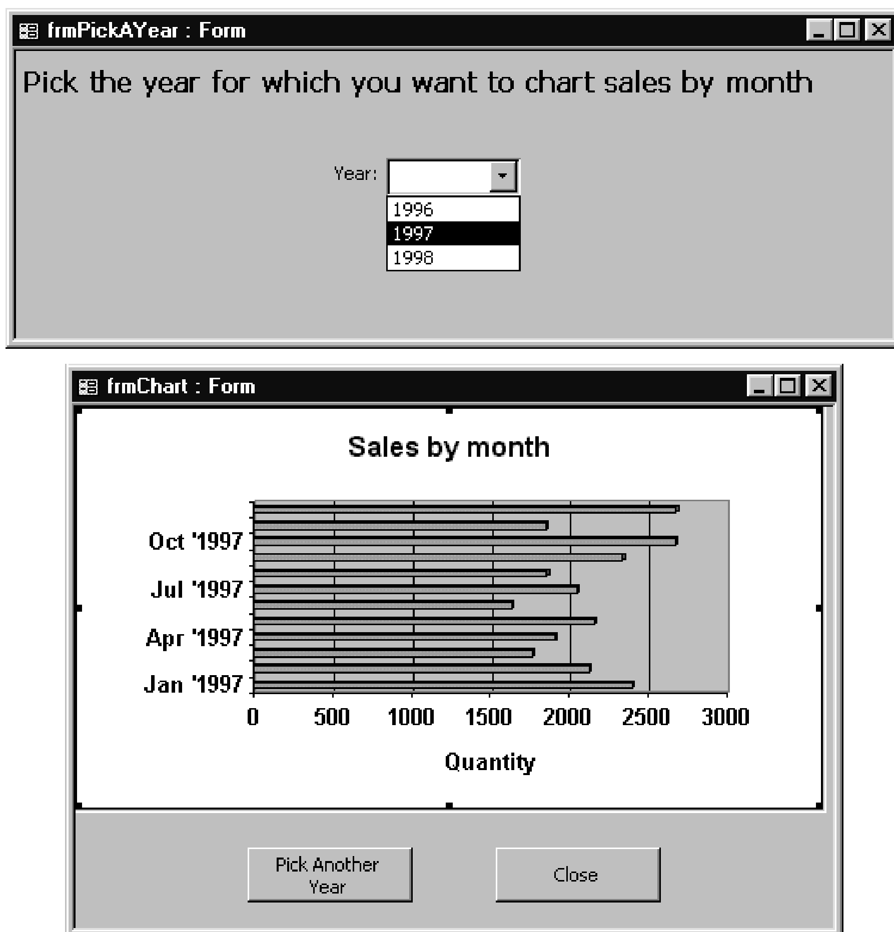
באפשרותך להוסיף אובייקט Chart באופן ידני, אך קל יותר לעשות זאת בעזרת אשף התרשימים. כל שעליך לעשות הוא לפעול כך:

1. לחץ על האובייקט **טפסים** (Forms) בחלון **מסד נתונים** (Database) ולאחר מכן לחץ על **חדש** (New).
2. בחר את **אשף התרשימים** (Chart Wizard) ואת הטבלה או השאילתה עליה ברצונך לבסס את התרשים. לאחר מכן לחץ על **אישור**.
3. בחר את השדות שיופיעו בגיליון נתוני התרשים ולאחר מכן לחץ על **הבא** (Next).
4. בחר את סוג התרשים ולחץ על **הבא**.
5. גרור ושחרר את לחצני השדה המבוקשים אל התרשים ולחץ על **הבא**.
6. בחר באפשרות **לשנות עיצוב של טופס או של תרשים** (Modify The Design Of The Form Or The Chart) ולחץ על סיום(Finish).

באפשרותך להוסיף פונקציות צבירה ועיצוב על ידי שינוי מחרוזת SQL במאפיין **מקור שורה** (Row Source) של פקד האובייקט הבלתי מאוגד המכיל את האובייקט Chart (האשף יוצר עבורך משפט זה).

הצגת התרשים באמצעות קוד

תרשים 5.14 מציג שני טפסים המאפשרים למשתמש להתוות את המכירות בחודש שנבחר. הטופס העליון מאפשר למשתמש לבחור שנה כלשהי מתוך הטבלה Orders. האירוע After Update של התיבה המשולבת פותח את הטופס התחתון המנצל את האובייקט Chart של Microsoft Graph 2000 כדי להציג את סך כל המכירות לפי חודשים בשנה שנבחרה.



תרשים 5.14: טפסים אלה מאפשרים למשתמש להציג נתוני מכירות חודשיים בשנה שנבחרה

התרשים שבתרשים 5.14 מבוסס על שאילתה שמאחזרת את כל ההזמנות של השנה שצוינה. השאילתה מתרגמת כל תאריך הזמנה לאחד בחודש (מבלי לשנות את נתוני הרקע). פעולה זו מאפשרת לצבור את כמות המכירות לפי חודש בצורה קלה ופשוטה, ועל ידי כך מאפשרת גם להתוות את המכירות לפי חודש (אשף התרשימים מסכם אוטומטית את כמות המכירות לפי חודש במקור רשומות כמו זה הנוכחי).

שלוש שגרות האירוע שלפניך שולטות באינטראקציה בין שני טפסים. השיגרה cboPickAYear_AfterUpdate טוענת את הטופס עם ציור התרשים וממזערת את הטופס שהמשתמש בוחר בו את השנה (עליך למזער את הטופס ולא לסגור אותו, מכיון ששאילתת התרשים קובעת את השנה שבחר המשתמש באמצעות התיבה המשולבת של הטופס הראשון).

```
Private Sub cboPickAYear_AfterUpdate()
    DoCmd.Minimize
    DoCmd.OpenForm "frmChart"
End Sub

Private Sub cmdClose_Click()
    DoCmd.Close acForm, "frmPickAYear"
    DoCmd.Close
End Sub

Private Sub cmdPickAYear_Click()
    DoCmd.OpenForm "frmPickAYear"
    DoCmd.Close acForm, "frmChart"
End Sub
```

טיפול בקבצים באמצעות VBA

בסעיף זה נתאר טכניקות אחדות למיכון השימוש בטפסים. נמחיש כיצד ממספרים טפסים ופקדים, כיצד מציגים ומסתירים טפסים באמצעות קוד וגם שיטות להצגת טופס בפרויקט אחר.

מספור טפסים ופקדים

האוסף AllForms, ששייך לאובייקט CurrentProject, מכיל פריט לכל טופס בפרויקט. יישום יכול למספר את האובייקטים מסוג AccessObject באוסף AllForms כדי למצוא את כל הטפסים בפרויקט. המאפיינים Name ו-IsLoaded של האובייקט AccessObject נוחים במיוחד לשימוש: המאפיין Name מייצג את השם של כל טופס בפרויקט, והמאפיין IsLoaded מציין אם הטופס פתוח.

השיגרה שלפניך מנצלת את המאפיין Count של האוסף AllForms כדי לקבוע את מספר הטפסים שבפרויקט הנוכחי. לאחר מכן היא מדפיסה את השם ואת מצב הטעינה של כל טופס בפרויקט (האוסף AllForms מאורגן כך שאינדקס החבר הראשון שלו הוא 0 ; לכן, הלולאה For עוברת על טפסי הפרויקט החל ב-0 ועד אחד לפני אחרון).

```
Sub ListAllForms()
Dim int1 As Integer
' Print the number of forms in the project.
   Debug.Print CurrentProject.AllForms.Count
   Debug.Print
' Enumerate each form in the project.
   For int1 = 0 To CurrentProject.AllForms.Count - 1
      Debug.Print CurrentProject.AllForms.Item(int1).Name
      Debug.Print CurrentProject.AllForms.Item(int1).IsLoaded
      Debug.Print
   Next int1
End Sub
```

האוסף AllForms מכיל את קבוצת כל הטפסים הפתוחים בפרויקט והאוסף Controls מכיל את קבוצת פקדי הטופס. היישומים שתיצור יכולים לנצל אוספים אלה כדי למצוא טופס מוגדר ופקד בודד בטופס זה.

השיגרה שלפניך ממספרת את כל הטפסים הפתוחים בפרויקט. היא מציגה את הפקדים של כל טופס פתוח לפי שם ולפי סוג. המאפיין ControlType של האובייקט Control מציין את סוג הפקד. באפשרותך להשתמש במילת המפתח TypeOf בדרך דומה.

```
Sub ListControlsOnOpenForms()
Dim frm1 As Form, ctl1 As Control
' Enumerate all open forms.
   For Each frm1 In Forms
      Debug.Print frm1.Name
' Enumerate each control on a specific open form.
      For Each ctl1 In frm1.Controls
         Debug.Print "     " & ctl1.Name & ", " & _
            IIf(ctl1.ControlType = 100, "label", "not label")
      Next ctl1
   Next frm1
End Sub
```

שים לב שהשיגרה מפענחת את ערך המאפין ControlType. כשערך זה הוא 100, הפקד הוא תווית. בתהליך הפענוח המעשי סביר יותר שתשתמש במשפט Select Case במקום בפונקציה Immediate If שבדוגמה הקודמת. הפונקציה Immediate If, לעומת זאת, פועלת היטב בפענוח ערך יחיד (באפשרותך לנצל את **Object Browser** (סורק האובייקטים) כדי למצוא את ערכי ControlType).

הסתרה והצגה של טפסים

באפשרותך להשתמש ב-VBA ובאוסף AllForms יחד עם אובייקטים נוספים כדי להסתיר את הטפסים בחלון מסד הנתונים. אם תהפוך את הטופס לבלתי נראה גם ביישום, המשתמש עלול לחשוב שסילקת את הטופס לחלוטין. טפסים מוסתרים יכולים עדיין לחשוף ערכים לשימוש באמצעות אובייקטים אחרים ביישום.

זוג השגרות שלפניך מסתיר ומבטל את הסתרת טופס של Access:

```
Sub HideAForm(frmName As String)

' Close form if it is open so that it can be hidden.
   If CurrentProject.AllForms(frmName).IsLoaded = True Then
      DoCmd.Close acForm, frmName
   End If
' Set form's Hidden property and do not show hidden
' objects in Database window.
   Application.SetHiddenAttribute acForm, frmName, True
   Application.SetOption "Show Hidden Objects", False
End Sub

Sub UnhideAForm(frmName As String)
' If form is hidden, set form's hidden property to False and open form.
   If Application.GetHiddenAttribute(acForm, frmName) = True Then
      Application.SetHiddenAttribute acForm, frmName, False
      DoCmd.OpenForm frmName
   End If
End Sub
```

השיטה SetHiddenAtrribute מסמנת או מנקה את המאפיין (Hidden) מחלון מסד הנתונים, של אובייקטי מסד נתונים, כגון טפסים, דוחות ושאילתות. שיטה זו קולטת שני ארגומנטים, אובייקט מסוג AccessObject וארגומנט בוליאני שמציין אם יש להסתיר את האובייקט. קריאה לשיטה באמצעות אובייקט ובצירוף הערך True זהה לקביעת המאפיין **מוסתר** (Hidden) של האובייקט בחלון מסד הנתונים.

השיטה SetHiddenAtrribute לבדה רק מעמעמת את האובייקט; המשתמשים יכולים עדיין לבחור אותו ולהשתמש בו. כדי להציג שאובייקטים מוסתרים יהיו בלתי נראים למשתמשים, בחר את **אפשרויות** (Options) מתפריט **כלים** (Tools), לחץ על **אובייקטים מוסתרים** (Hidden Objects) ולאחר מכן לחץ על **אישור**.

בטרם תפעיל את השיטה SetHiddenAtrribute, עליך לבדוק את המאפיין IsLoaded של האובייקט AccessObject. אם האובייקט טעון, עליך לסגור אותו בטרם תנסה להפעיל את SetHiddenAtrribute; קריאה לשיטה כשהאובייקט פתוח גורמת לשגיאה.

מספור טפסים בפרויקט אחר

VBA אינו מגביל לטיפול באובייקטים של מסד נתונים בפרויקט הנוכחי בלבד. לדוגמה, באפשרותך לבדוק קיום של טפסים ביישום אחר של Access. אחד השלבים החיוניים בתהליך זה הוא להשוות את המאפיין Name (שם) של כל חברי האוסף AllForms לשם טופס היעד. קיימת תחבולה נוספת שעליך ללמוד: פותחים מופע חדש של טופס Application של Access כשהוא כולל את מסד נתוני היעד ולאחר מכן משתמשים בפרויקט הנוכחי של מופע זה בתור המקור עבור האוסף AllForms. שיפור עדין זה מאפשר לטפל באובייקטי מסד נתונים השייכים לקובץ מסד נתונים אחר.

שתי השגרות שלפניך מממשות את הרעיון באמצעות VBA. השיגרה FormToLookFor מגדירה את נתיב מסד הנתונים אל קובץ מסד הנתונים האחר וקולטת את השם של טופס היעד. השיגרה השנייה, FormExistsInDB, מחפשת טופס יעד. באפשרותך לקרוא לטופס השני מתוך הטופס הראשון.

```
Sub FormToLookFor()
Dim strDB As String
Dim strFormName As String
' Search for forms in the Northwind database.
   strDB = "C:\Program Files\Microsoft Office\Office\Samples\Northwind.mdb"
' Get the name of the form to search for from the user.
   strFormName = InputBox("Enter the name of the form to search for: ", _
      "Programming Microsoft Access 2000")

' Call FormExistsInDB to check whether the form exists.
   FormExistsInDB strDB, strFormName
End Sub

Sub FormExistsInDB(strDB As String, strFormName As String)
Dim appAccess As Access.Application, int1 As Integer

' Return reference to Microsoft Access Application.
   Set appAccess = New Access.Application

' Open a database in the other application.
   appAccess.OpenCurrentDatabase strDB

' Check whether the form exists.
   For int1 = 0 To (appAccess.CurrentProject.AllForms.Count - 1)
      If (appAccess.CurrentProject.AllForms.Item(int1).Name _
         = strFormName) Then
         MsgBox "Form " & strFormName & " exists in the " & strDB & _
            " database.", vbInformation, "Programming Microsoft Access 2000"
         GoTo FormExistsExit
      End If
   Next int1
```

```
' Report that form does not exist.
   MsgBox "Form " & strFormName & " does not exist in the " _
      & strDB & " database."

' Close other Access application.
FormExistsExit:
   appAccess.CloseCurrentDatabase
   Set appAccess = Nothing
End Sub
```

השיגרה הראשונה נותנת ל-strDB ערך טיפוסי של נתיב אל מסד הנתונים Northwind. אם קיים עותק של Northwind במקום אחר כלשהו, עליך לעדכן את הנתיב. פונקציה מסוג InputBox מנחה את המשתמש להזין את שם הטופס שיש לחפש, ולאחר מכן השיגרה הראשונה קוראת לשיגרה השנייה.

השיגרה השנייה קובעת ופותחת הפניה אל המופע החדש של היישום של Access ולאחר מכן מגיעה ללולאה שבודקת אם טופס כלשהו במסד הנתונים החדש תואם לשם טופס היעד. השיגרה מדווחת אם מצאה את טופס היעד ומשחררת את המשאבים שהוקצו למסד הנתונים החדש לפני חזרתה.

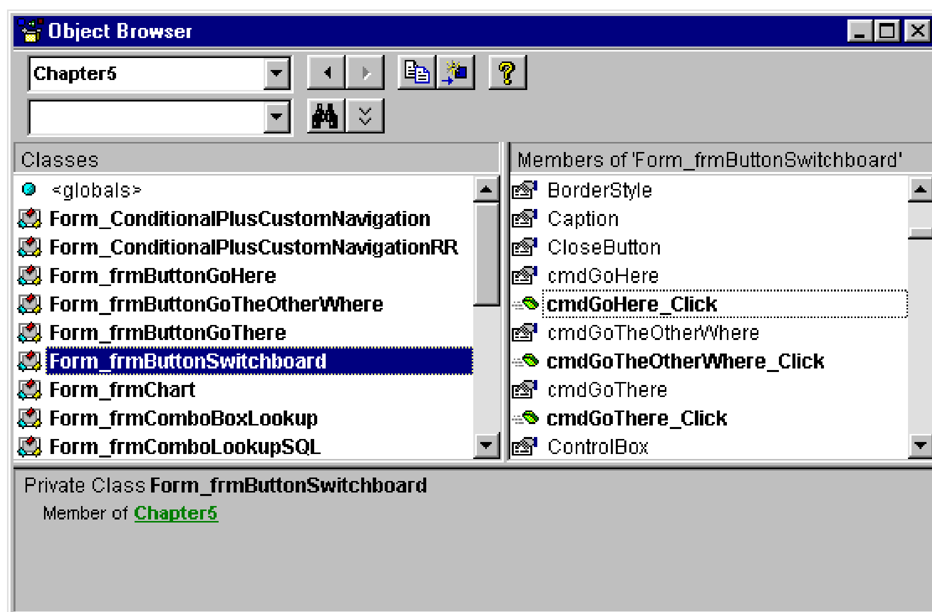
שימוש במחלקות טופס

טופס כלשהו של Access 2000 בעל מודול מאחוריו הוא מחלקת טופס; באפשרותך ליצור מופעים חדשים של המחלקה בעזרת מילת המפתח New בדיוק כפי שהיית נוהג עם מחלקה כללית (גנרית) כלשהי של Access. אחד היתרונות של מחלקות טופס הוא שהן כוללות את כל השיטות ומאפייני הטופס התקניים וגם את התוספות המותאמות אישית שהמְפַתח מוסיף להן.

הצגת מאפיינים ושיטות של מחלקת טופס

באפשרותך להציג את המאפיינים והשיטות של מחלקות טופס באמצעות **Object Browser** (סורק האובייקטים). בחר בשם הפרויקט בתיבת הרשימה הנפתחת Project/Library (פרויקט/ספריה) ובחר שם של מחלקת טופס כדי להציג את המאפיינים והשיטות שלו.

סורק האובייקטים שבתרשים 5.15 מציג קבוצת משנה של חברים במחלקה Form_frmButtonSwitchboard. כזכור, טופס זה מנווט אל שלושה טפסים אחרים באמצעות שגרות אירוע. שגרות האירוע הן שיטות מחלקה, כגון cmdGoHere_click. הלחצנים, כגון cmdGoHere, הם מאפייני מחלקה.



תרשים 5.15: שיטות ומאפיינים של מחלקת טופס

טיפול במחלקת טופס

השיגרה שלפניך מפנה למודולי מחלקה ב- Access 2000. השיגרה כוללת כמה מקטעי קוד שמטפלים במחלקת הטופס הבסיסית בדרכים מתוחכמות ומתקדמות יותר. הטיפול במחלקת טופס ובמופעים שלהן דומה לטיפול בעוגיות באמצעות חותך עוגיות. מכשיר החיתוך הוא מחלקת הטופס והעוגיות הן מופעים שלו. שינוי בעוגיה אינו משפיע על מכשיר החיתוך. לעומת זאת, שינויים במכשיר החיתוך משפיעים על כל העוגיות לאחר ביצוע השינוי.

```
Sub testformclass()
Dim frm1 As Form

' First code segment
' Saves reference to instance of a form class in frm1.
' Can reference with either class or reference name.
   Set frm1 = Form_frmCustomers
   frm1.Caption = "foo"
   MsgBox Form_frmCustomers.Caption
   MsgBox frm1.Caption
   DoCmd.Close acForm, frm1.Name
```

```
' Second code segment
' Programmatically alters and opens default form instance.
' Does not set reference to instance.
' Clears instance by setting Visible to False.
    Form_frmCustomers.Caption = "foo"
    Form_frmCustomers.Visible = True
    MsgBox Form_frmCustomers.Caption
    If MsgBox("Do you want to close form instance?", vbYesNo, _
        "Programming Microsoft Access 2000") = vbYes Then
        Form_frmCustomers.Visible = False
    End If

' Third code segment
' Open form in Design view to modify class properties.
' Open in Form view to see impact of Design view change.
    DoCmd.OpenForm "frmCustomers", acDesign
    Forms("frmCustomers").Caption = "foo"
    MsgBox Form_frmCustomers.Caption
    DoCmd.Close acForm, "frmCustomers", acSaveYes
    Form_frmCustomers.Visible = True
    MsgBox Form_frmCustomers.Caption

' Fourth code segment
' Restore class caption property.
    DoCmd.OpenForm "frmCustomers", acDesign
    Set frm1 = Form_frmCustomers
    frm1.Caption = "Customers"
    DoCmd.Close acForm, frm1.Name, acSaveYes
    Form_frmCustomers.Visible = True
    MsgBox Form_frmCustomers.Caption

End Sub
```

מקטע הקוד הראשון מקצה מחלקת טופס להפניה. הוא יוצר מופע של המחלקה Form_frmCustomers ומקצה אותו ל-frm1. באפשרותך לטפל במופע באמצעות שם המחלקה או שם ההפניה; התוצאות הזהות שמתקבלות משתי תיבות ההודעה מאשרות זאת. שינוי המופע של מחלקת טופס אינו משנה את המחלקה עצמה.

אין צורך בהפניות כדי לציין או לשנות את המאפיינים של מופעי מחלקת הטופס. המקטע השני מבצע משימה זהה למשימה הראשונה מבלי ליצור מצביע אל מחלקת הטופס. מקטע זה כולל גם הנחיה ששואלת את המשתמש אם ברצונו לסגור את מופע הטופס (מופע מתקיים לאורך חיי השיגרה שיוצרת אותו, אלא אם הקוד מסיים אותו קודם לכן).

מקטע הקוד השלישי פותח את הטופס בתצוגת העיצוב ומטפל במאפייניו. בניגוד למקרה בו משנים את מאפייני הטופס באמצעות קוד בתצוגת **טופס** (Form), שינויי מאפיינים שנערכו בתצוגת העיצוב ממשיכים להתקיים גם לאחר שמירת הטופס. שתי השורות האחרונות של המקטע השלישי פותחות מופע של מחלקת הטופס בתצוגת **טופס**. בפעם הראשונה בשיגרה, הטופס נפתח עם הכיתוב foo.

מקטע הקוד הרביעי מחזיר את כיתוב מחלקת הטופס לערכו המקורי, "Customers".

הפניות למופעי מחלקת טופס

דוגמת מחלקת הטופס שלפניך יורדת לעומקן של מחלקות טופס. באפשרותך לבדוק בקלות את התנהגות השיגרה על ידי שימוש בטופס frmFirstWithControls, הכולל לחצן שמפעיל את השיגרה testformclass2. שיגרה זו מטפלת במופעים רבים של מחלקת טופס.

```
Sub testformclass2()
On Error GoTo testclass2Trap
Dim frm1 As Form
Dim frm2 As New Form_frmCustomers
Dim int1 As Integer
' Show caption of frm2 before editing the property.
    MsgBox "Frm2 default caption is " & frm2.Caption, _
        vbInformation, "Programming Microsoft Access 2000"
' Set frm1 as a reference to frmCustomers class.
    Set frm1 = Form_frmCustomers
    frm1.Caption = "Caption from frm1 instance"
' Reset caption for frm2 instance.
    frm2.Caption = "Caption from frm2 instance"

' Show the captions of the class instances referenced by frm1 and frm2.
    frm1.SetFocus
    frm2.SetFocus
    MsgBox frm1.Caption, vbInformation, "Programming Microsoft Access 2000"
    MsgBox frm2.Caption, vbInformation, "Programming Microsoft Access 2000"

' Close form instances by their references.
    frm1.SetFocus
    DoCmd.Close
    frm2.SetFocus
    DoCmd.Close

testclass2Exit:
    Exit Sub
```

```
testclass2Trap:
    If Err.Number = 2467 Then
' Trap attempt to print caption for closed form.
        MsgBox "Cannot print caption of closed form", _
            vbInformation, "Programming Microsoft Access 2000"
        Resume Next
    Else
        Debug.Print Err.Number, Err.Description
        Resume testclass2Exit
    End If

End Sub
```

נוח מאוד להשתמש בהפניות אל מופעי מחלקת טופס בעת שמטפלים ביותר ממופע אחד. הדוגמה לעיל מנצלת את המופעים frm1 ו-frm2. לאחר ההצהרה על frm1 בתור מחלקת טופס כללית, הקוד מקצה לו בהמשך הפניה אל מופע של המחלקה Form_ frmCustomers. משפט ההצהרה על frm2 יוצר הפניה אל מופע של אותה מחלקת טופס, ולכן frm1 ו-frm2 הם שני מופעים נפרדים של מחלקת טופס זהה.

משפט Dim השני יוצר מופע המבוסס על מחלקת טופס, באמצעות מילת המפתח New (עליך להתייחס אל מחלקת טופס בעלת התחילית Form_ בעת שימוש במילת המפתח New ליצירת מופע). לאחר יצירת ההפניות תוכל להפעיל שיטות ומאפייני טופס רגילים של אובייקטים שאליהם מתייחסים המשתנים. לדוגמה, השיגרה מגדירה את המאפיין Caption של frm2 ולאחר מכן מעבירה את המיקוד ראשית אל frm1 ולאחר מכן אל frm2. על ידי כך נפתחים שני הטפסים זה על גבי זה במסך של Access.

באפשרותך לסגור מופע של טופס על ידי העברת המיקוד אליו ולאחר מכן להפעיל את השיטה Close של האובייקט DoCmd.

6

יצירת דוחות

הידע שרכשת בפרק הקודם בנושא טפסים, יהיה לך לעזר רב כשתרצה לשלב דוחות ביישומים שתיצור. למרות שדוחות אינם תומכים בפקדים אינטראקטיביים, תוכל לאכלס דוחות בפקדים שמציגים נתונים, כגון תיבות טקסט ותיבות סימון. דוחות יכולים להכיל גם תמונות גרפיות מאוגדות ובלתי מאוגדות ופקדי ActiveX לתצוגות מיוחדות, כגון תרשימים. ב- Microsoft Access 2000 היכולת ליצור תרשים בדוח פועלת בצורה דומה מאוד ליכולת המקבילה בטפסים.

בפרק זה נעסוק בהיבטים מעשיים של עיצוב ותצוגה בדוח. נעסוק בתכנות להצגת תוכן דינמי, וגם במספור ובטיפול בדוחות ובפקדים. הפרק מציג מקרה לדוגמה של ספר אורחים ב-ProntPage המיועד לעסק קטן שלו צרכים מחלקתיים שבדרך כלל ניתנים לטיפול ב-Access. שני סעיפים בפרק דנים בעיצוב דוחות. סעיף אחר עוסק בהצגת דוחות באמצעות תבנית קובץ תמונה (Snapshot) של Access. תבנית זו מאפשרת לשתף דוחות Access באמצעות דואר אלקטרוני באינטרנט, גם בתחנות עבודה שאין בהן Access. שלושת הסעיפים האחרונים מציגים דוגמאות בנושאי תכנות ספציפיים. שני הסעיפים הראשונים עוסקים בתוכן דינמי בדוח. הסעיף האחרון מתעמק בטיפול באמצעות קוד בדוחות ובפקדים. סעיף זה מציג טכניקות לטיפול בסוגים רבים של האובייקט AccessObject, כגון חברים באוספים AllReports ו-AllForms.

כיצד ליצור דוח

תוכל ליצור דוח בשלוש דרכים:

- שימוש באשף הדוחות האוטומטיים (AutoReport Wizard). זו הדרך הפשוטה ביותר, שאינה דורשת יותר מלחיצות אחדות בעכבר.
- שימוש באשפים אחרים. אשף הדוחות (Report Wizard) חוסך זמן ואילו אשף התוויות (Label Wizard) ואשף התרשימים (Chart Wizard) מסייעים להתמודד עם בעיות עיצוב מיוחדות.
- שימוש בתצוגת **עיצוב** (Design). יצירת דוחות ידנית בתצוגת עיצוב מאפשרת גמישות מירבית בכל הנוגע לפריסה, עיצוב, מיון וקיבוץ.

שימוש באשף הדוחות האוטומטיים

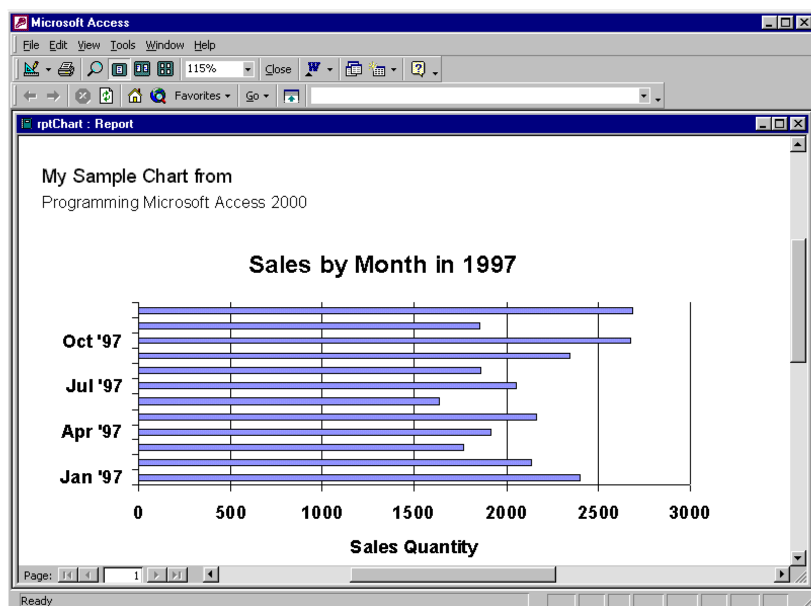
דוח אוטומטי המופעל מסרגל הכלים שבחלון מסד הנתונים, הוא דוח ערוך בטורים. שדות טקסט ושדות מספריים מוצגים בתיבות טקסט, נתונים מסוג **כן/לא** (Yes/No) מוצגים בתיבות סימון ונתונים מסוג **אובייקט OLE** (OLE Object) (כגון תמונות מסוג *.bmp) מוצגים במסגרות אובייקטים מאוגדים. הרשומות מוצגות בזו אחר זו בתבנית זורמת, החל ברשומה הראשונה וכלה באחרונה.

תוכל גם לבחור דוח טבלאי בתיבת הדו-שיח **דוח חדש** (New Report). משיקולי מהירות ונוחיות, מפתחים מתקדמים וכאלה בעלי ניסיון ממוצע יעדיפו ליצור דוח משנה באופן זה.

שימוש באשפים נוספים

Access כולל אשף דוחות כללי וגם אשפים מיוחדים ליצירת תוויות ותרשימים. האשפים מאפשרים ליצור דוחות וסוגי אובייקטים נוספים של מסד נתונים. אשפים מסוגלים להאיץ את תהליך הפיתוח ולהנהיג סטנדרטיזציה שמפשטת את התחזוקה.

אשף הדוחות (Report Wizard) מאפשר להגדיר דוח בצורה דומה מאוד לזו שבה אשף השאילתות הפשוטות מסייע בעיצוב שאילתה (ראה פרק 4). האשף חוסך את הזמן והמאמץ בעיצוב השאילתה כשלב מקדמי להכנת דוח. בנוסף, הוא מפשט פעולות כגון קביעת סדר מיון, קיבוץ ועיצוב דוחות מותאמים אישית. **אשף התוויות ואשף התרשימים** מבצעים פעולות מיוחדות. המקרה לדוגמה של ספר האורחים שיוצג בהמשך הפרק, מציג דוגמת פלט של אשף התוויות. אשף התרשימים של דוחות דומה בפעולתו לאשף התרשימים של טפסים. תרשים 6.1 מציג דוח תרשים במצב הצג לפני הדפסה (כדי לרענן את זיכרונך בנושא אובייקט Chart של Microsoft Graph 2000, עיין בפרק 5 בקטע העוסק בהעברת קבוצת משנה של נתונים לתרשים).



תרשים 6.1: תרשים זה הוכן באמצעות אשף התרשימים של דוחות

יצירה ידנית של דוח בתצוגת עיצוב

תוכל ליצור דוח בצורה ידנית על ידי לחיצה כפולה על האפשרות **יצירת דוח בתצוגת עיצוב** (Create Report In Design View) בחלון מסד הנתונים (תחילה בחר את **דוחות** (Reports) מהאוסף **אובייקטים**). ייפתח דוח ריק שמאפייני הבסיס שלו משקפים את הערך שבתיבת הטקסט **תבנית דוח** (Report Template) בכרטיסיה **טפסים/דוחות** (Forms/Reports) של תיבת הדו-שיח **אפשרויות** (Options) (פותחים את תיבת הדו-שיח באמצעות הפקודה **כלים** (Tools), **אפשרויות**). ברירת המחדל היא **רגיל** (Normal), והיא מציגה דף לבן פשוט ללא צבעים או גופנים מיוחדים. אם יש לך דוח שברצונך לנצל את הגדרות המאפיינים שלו כתבנית לכל שאר הדוחות בפרויקט Access, תוכל להכניס את שם הדוח לתיבת הטקסט **תבנית דוח**. הגדרות דוח זה תהיינה הגדרות ברירת המחדל של כל הדוחות החדשים. דוחות קיימים יישארו ללא שינוי. תוכל לנצל את התבנית כדי לקבוע את מספר המקטעים שיוצגו בהתחלה בדוח וגם את הגדרות ברירת המחדל של מאפייני מקטע והפקדים.



הערה:

תיבת הטקסט **תבנית טופס** (Form Template) משמשת ליצירת טפסים חדשים כפי ש**תבנית דוח** משמשת ליצירת דוחות.

לאחר שתפתח מופע דוח חדש המבוסס על תבנית, תוכל לאכלס אותו בפקדים. חשוב להבין כי דוחות Access ערוכים במקטעים. בתבנית הרגילה למשל, המקטע **פירוט** (Detail) מוצג בין זוג מקטעי העמוד. כל תוכן הכלול במקטע העליון והמקטע התחתון מופיע בחלק העליון והתחתון בכל עמוד בדוח. המקטע **פירוט** חוזר על עצמו אחת לכל רשומה בעמוד, וגוזר ערכים עוקבים מתוך מקור הרשומות של הדוח.

זוג מקטעים נוסף שיכול לתחום מקטע **פירוט** בדוח הם המקטעים **כותרת עליונה של דוח** (Report Header) **וכותרת תחתונה של דוח** (Report Footer). מקטעים אלה מופיעים פעם אחת בלבד בתחילת הדוח ובסופו. מפעילים ומכבים את המקטעים **דוח** ו**עמוד** התוחמים את המקטע המרכזי **פירוט**, באמצעות תפריט **תצוגה** (View).

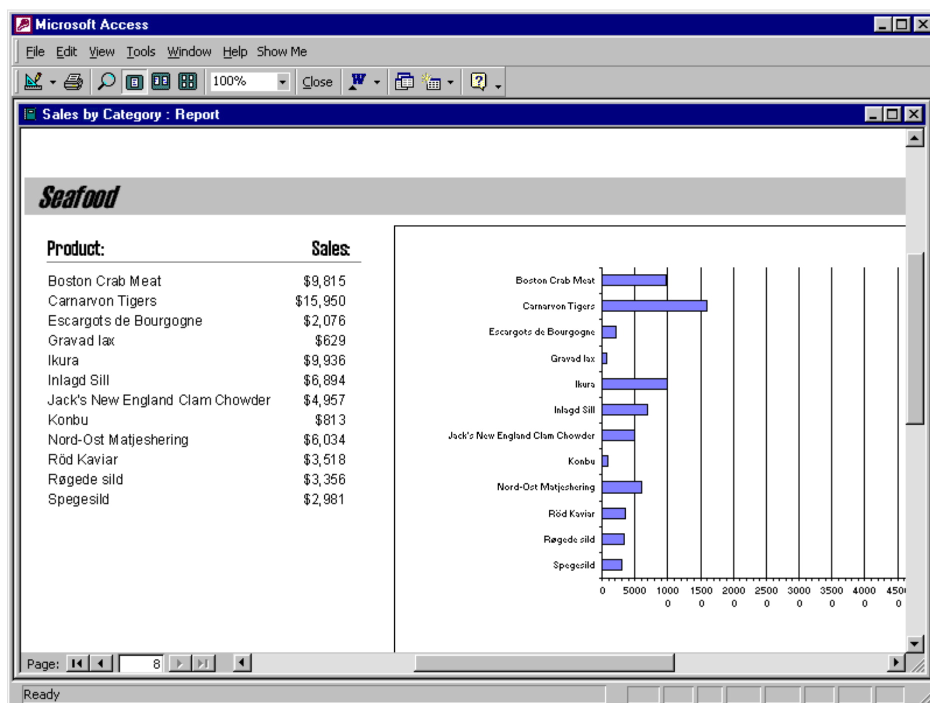
באפשרותך לשנות את גודלו של מקטע דוח כלשהו על ידי גרירה של גבולותיו. לדוגמה, כדי להציג מקטע כותרת עליונה של דוח מבלי להציג את מקטע הכותרת התחתונה שלו, צמצם את המקטע לאפס, על ידי גרירת את הגבול התחתון של מקטע הכותרת התחתונה עד שיישק לגבול העליון שלו.

הדוח שבתרשים 6.1 מכיל את המקטע **כותרת עליונה של עמוד** (Page Header) ואת המקטע **פירוט**. המקטע **כותרת עליונה של עמוד** מכיל שתי כותרות: My Sample Chart ו- Programming MS Access 2000 from. מסגרת האובייקט הבלתי מאוגד מוצגת מתוך המקטע **פירוט** של הדוח. פקד המסגרת אינו מאוגד למקור הרשומות, ולכן הוא מופיע פעם אחת בלבד.

פקדי דוח

דוחות של Access מכילים באופן טיפוסי תמהיל של תיבות טקסט ותוויות. תיבות הטקסט משקפות את ערכי השדות של מקור הרשומות שברקע, והתוויות מזהות את השמות של תיבות הטקסט ושל פקדים נוספים. פקדים נוספים שמציגים ערכי שדות בדוח כוללים תיבות סימון ומסגרות אובייקטים מאוגדים. המסגרות הן אמצעי הולם לשמירת תמונות גרפיות בטבלה. פקדי תמונה משמשים לשמירת תמונות בלתי מאוגדות שאינן ערכי שדות בטבלה.

דוח ראשי יכול לשמש גם בתור מארח של דוח משנה אחד או יותר שמופיעים כפקדים בדוח הראשי. הדוח Sales By Category במסד הנתונים Northwind (תרשים 6.2) מכיל דוח משנה שמציג מכירות לפי מוצר. הדוח הראשי מציג נתונים זהים אך בצורה גרפית. דוח המשנה מופיע לשמאלו בצורת טבלה. התרשים מנצל את האובייקט Chart של Microsoft Graph 2000 במסגרת בלתי מאוגדת. דוח המשנה והתרשים מופיעים שניהם במקטע הכותרת העליונה CategoryName, מה שגורם לדוח המשנה ולתרשים לבחור את המוצרים השייכים לשם הקטגוריה המופיע בכותרת המקטע בכל דף. כל דוח יכול להכיל מקטעי קבוצה מקוננים, כגון מחוזות במדינות או סוגי מוצרים בחטיבות של חברה.



תרשים 6.2: דוח משנה ותרשים בדף הדוח Sales By Category. המוצרים משתנים בכל בכל כותרת של קטגוריה.

קוד ברקע של דוחות

יישומים מותאמים אישית דורשים בדרך כלל דוחות ידניים המעוצבים באופן שהולם תכנים מוגדרים. Access מציע מיגוון רחב של אפשרויות עיצוב באמצעות קוד, אך קרוב לוודאי שתנצל רק חלק קטן מהן:

- הדפסה או הצגה מקדימה אוטומטית של דוח על ידי הפעלת השיטה DoCmd OpenReport, או יצירת היפר-קישור לדוח (סעיף "טופס מסך ניווט" בפרק 5 מציג דוגמאות שניתן לאמץ בדוחות).
- הפיכת הדוח לדינמי. ניתן לעדכן מאפיין אחד או יותר של האובייקט Report. לדוגמה, באפשרותך לשנות את מאפיין מקור הרשומות של דוח ואת הכיתוב של תוויות בדוח.
- מספור חברי האוסף AllReports כדי לקבוע אם דוח קיים ואם הוא פתוח.
- יצירת מופעים רבים של דוחות (דוחות בעלי מודולים הם מודולי מחלקה).

מפתחים מנוסים וכאלה בעלי ניסיון ממוצע עובדים לעיתים קרובות עם אירועי דוח, הרגל שתוכל לאמץ כדי לשלוט בצורה דינמית בתוכן הדוח ובעיצובו. השתמש באירועי הדוח המותאמים אישית Open, Activate, Close ו-Deactivate במקום באירועים

Initialize ו-Terminate לכל מופע חדש של מודול המחלקה שקשור אל דוח. בעת הדפסה או הצגה לפני הדפסה של דוח, מתרחשים האירועים Format ו-Print כנגד המקטעים העוקבים שבדוח. שני האירועים הללו שימושיים לשינוי דינמי של תוכן הדוח ועיצובו. הגדרת המאפיין Keep Together בתור True עלולה לגרום לאירוע Retreat להתרחש בצורה סלקטיבית. מקטעי דוח מפצלים רצועות דוח למרכיבים תואמים אך נבדלים זה מזה. לדוגמה, המקטע **כותרת עליונה של דוח** (Report Header) הוא המקטע הראשון כשהוא מוצג בדוח, והמקטע **כותרת תחתונה של דוח** (Report Footer) הוא המקטע האחרון כשמקטע זה כלול בדוח.

מקרה לדוגמה:
ספר האורחים של FrontPage

FrontPage מאפשר ליצור בקלות ספר אורחים המרכז מידע על מבקרים (כגון שם ופרטי יצירת קשר) בדף Web בתבנית HTML. גישה זו אינה מחייבת ידע כלשהו במסדי נתונים או בחיבורי קישוריות מסד נתונים פתוחה (ODBC). מבקרי האתר רואים טופס בעל חזות מקצועית שאליו הם יכולים להוסיף פרטי איש קשר, ומשתמשי FrontPage יכולים ליצור ספר אורחים ולהציגו בצורה קלה ופשוטה. השיטה מתפקדת היטב כל עוד מספר המבקרים באתר אינו רב מדי.

אם כמות המבקרים באתר גדלה משמעותית, ניתן להעביר את כל נתוני אנשי הקשר לקובץ מסד נתונים. גם אם האתר מעביר את הנתונים להפקדה במסד נתונים, עדיין נותרה הבעיה לשחזר את המידע המקורי המעוצב ב-HTML. מקרה זה מדגים גישה אחת לשחזור נתוני טקסט ישנים ומתאר כיצד להפיק תוויות מען וטפסים המכילים את המידע לאחר המרתו לתבנית המתאימה.

ייבוא נתונים

ניתן לנצל את אשף יבוא טקסט כדי להעתיק את קובץ ספר האורחים הגולמי אל טבלה של Access. תרשים 6.3 מציג חלקים ובהם שתי רשומות מתוך קובץ HTML הגולמי (עבור האורחים Karl Doe1 ו- Boban Doe2). שים לב למספר העצום של תגי HTML. הסייר הפנימי של FrontPage מנצל תגים אלה כדי לעצב את תצוגת ספר האורחים, אך התגים אינם מאפשרים ייבוא נתונים ישירות לטבלה של Access להכנת תוויות מען. כל מידע אנשי הקשר מופיע בעמודה ארוכה יחידה. יש שורות שמכילות נתוני קשר, אך שורות אחרות מכילות תוויות תיאוריות או תווי עיצוב כלליים. כללי העיבוד של מסדי נתונים יחסיים מחייבים להקצות שורה נפרדת לכל אורח שנרשם בספר האורחים. הדבר מחייב לפזר את המידע של אורח אחד הרשום בעמודה יחידה, על פני כמה עמודות.

```
Temp1 : Table
Field1
<dl>
<dt><b>SiteEvaluation_FirstName: </b></dt>
<dd>Karl</dd>
<dt><b>SiteEvaluation_LastName: </b></dt>
<dd>Doe1</dd>
<dt><b>SiteEvaluation_Organization: </b></dt>
<dd>HLC of Virginia</dd>
<dt><b>SiteEvaluation_StreetAddress: </b></dt>
<dd>6221 Barrister Rd</dd>
<dt><b>SiteEvaluation_Address2: </b></dt>
<dd> </dd>
<dt><b>SiteEvaluation_City: </b></dt>
<dd>Chesterfield</dd>
<dt><b>SiteEvaluation_State: </b></dt>
<dd>va</dd>
<dt><b>SiteEvaluation_ZipCode: </b></dt>
<dd>23832</dd>
<dt><b>SiteEvaluation_Country: </b></dt>
<dd>US</dd>
<dt><b>SiteEvaluation_FAX: </b></dt>
<dd> </dd>
<dt><b>SiteEvaluation_Email: </b></dt>
Record: 12 of 4295
```

```
Temp1 : Table
Field1
<dl>
<dt><b>SiteEvaluation_FirstName: </b></dt>
<dd>Boban</dd>
<dt><b>SiteEvaluation_LastName: </b></dt>
<dd>Doe2</dd>
<dt><b>SiteEvaluation_Organization: </b></dt>
<dd>Technical School &quot;Nikola Tesla&quot;</
<dt><b>SiteEvaluation_StreetAddress: </b></dt>
<dd>Tabacka carsija 9/10</dd>
<dt><b>SiteEvaluation_Address2: </b></dt>
<dd> </dd>
<dt><b>SiteEvaluation_City: </b></dt>
<dd>Pozarevac</dd>
<dt><b>SiteEvaluation_State: </b></dt>
<dd>Serbia</dd>
<dt><b>SiteEvaluation_ZipCode: </b></dt>
<dd>12000</dd>
<dt><b>SiteEvaluation_Country: </b></dt>
<dd>Yugoslavia</dd>
<dt><b>SiteEvaluation_FAX: </b></dt>
<dd>2345678901</dd>
<dt><b>SiteEvaluation_Email: </b></dt>
Record: 49 of 4295
```

תרשים 6.3: קטעים מקובץ ספר האורחים של FrontPage, שיובא אל טבלה של Access

בין אם נתקלת בדרישת המרת נתונים מדויקת זו ובין אם לאו, פיתוח ב-Access מחייב באופן טיפוסי לטפל בזרימת טקסט. עסקים גדולים נדרשים לעיתים קרובות להמיר נתונים מחלקתיים ישנים לתבנית של מסד נתונים יחסי באמצעות תהליך דומה. המיגוון העשיר של פונקציות עיבוד מחרוזת והשילוב של Visual Basic for Applications (VBA) ב-Access מאפשרים להפוך משימה זו לאוטומטית.

המרת מבנה הנתונים

המטרה של עיבוד מחרוזת היא להפוך את הנתונים שבטבלה כגון Temp1 שבתרשים 6.3 לטבלת אנשי קשר מקובלת, כגון זו שבתרשים 6.4. אסטרטגיית הפעולה כוללת שתי ערכות רשומות: אחת עבור טבלת המקור ואחת עבור טבלת היעד. השיגרה עוברת לאורך שורות טבלת המקור כדי לחלץ את נתוני אנשי הקשר ולאחסן את הנתונים שעברו ניתוח במשתנים. לאחר שסיימה לעבד את כל הנתונים של איש קשר, השיגרה מוסיפה אותם לשורה חדשה בטבלת היעד. מכיון שלשדות השונים של איש קשר יש דרישות ייחודיות, חוקי העיבוד משתנים לעיתים משדה לשדה. השינוי בחוקים עשוי לנבוע מכך שנתונים גולמיים בשדה גורמים לבעיות ייחודיות שלא קורות בשדות אחרים.

קטע השיגרה הבא הופך את תבנית נתוני הטבלה שבתרשים 6.3 לתבנית נתוני הטבלה בתרשים 6.4. הנוסח המלא של השיגרה מופיע בתקליטור המצורף לספר. הגירסה המקוצרת שבפרק מציגה את הקוד שמבצע את המרת שלושת השדות הראשונים המיוצגים באמצעות המשתנים strFname, strLname ו-strCname. תדפיס הקוד ארוך והנוסח המלא שלו ארוך עוד יותר, אך המרת נתונים היא השלב הראשון הקריטי של פרויקטים של מסדי נתונים של Access. אם לא תצליח לטפל בנושא קריטי זה, פרויקט Access שלך לא יפעל.

WebBasedList : Table

Contact ID	First Name	Last Name	Company Name	Address	Address	City
2	Karl	Doe1	HLC of Virginia	6221 Barrister Rd		Chesterfielc
3	Boban	Doe2	Technical School 'Nikola Tesla'	Tabacka carsija 9/10		Pozarevac
4	Quilla	Doe3	Enterprise Connection	22 Willow St.		Marysville
5	Steven	Doe4	Apostolic Faith Church	3823 S. Indiana Ave		Chicago
6	Garland	Doe5	Bassett Furn Ind Inc	662 Westover Dr		Bassett
7	Conrad	Doe6	Saddleback Community College	2214-B Via Mariposa E		Laguna Hill:
8	Jason	Doe7	Spydee Internet Serivces	2240 S. Hwy 29		Pensacola
9	Bill	Doe8	Digital Listing Services	2201 Baker Court		Antioch
10	John	Doe9	Powermatic Sdn Bhd	71 Jln USJ 11/1g		Subang Jay
11	ADIL	Doe10	RESEARCH	31-77	36 STREET	ASTORIA

Record: 1 of 101

תרשים 6.4: נתוני ספר האורחים של FrontPage לאחר המרה מתבנית HTML. ההמרה מסירה את תגי העיצוב ומציגה כל רשומת איש קשר בשורה נפרדת.

```
Sub getfp()
Dim cnn1 As New ADODB.Connection
Dim rst1 As ADODB.Recordset
Dim strFname As String, strLname As String
Dim strCname As String, strSt1 As String
Dim strSt2 As String, strCity As String
Dim strSt As String, strPostal As String
Dim strCountry As String, blSkip As Boolean
Dim rst2 As New Recordset
' Open two recordsets and set references to them.
    cnn1 = CurrentProject.Connection
    Set rst1 = New Recordset
    rst1.ActiveConnection = CurrentProject.Connection
    rst1.CursorType = adOpenKeyset
    rst1.LockType = adLockOptimistic
' Raw contact information is in table temp1.
    rst1.Open "temp1"
    rst2.ActiveConnection = CurrentProject.Connection
    rst2.CursorType = adOpenKeyset
    rst2.LockType = adLockOptimistic
' The application stores parsed contact info in the WebBasedList table.
    rst2.Open "WebBasedList"

' Start a loop through the recordset of raw contact information.
    Do Until rst1.EOF
        blSkip = False
' Start a new contact record when you find
'a label named "SiteEvaluation_FirstName:".
        If InStr(1, rst1.Fields(1),"SiteEvaluation_FirstName:") <> 0 Then
            rst1.MoveNext
            If rst1.Fields(1) <> "  <dd> </dd>" Then
```

```
' The length of the first name field is the number of
' characters between ">" and "<" delimiters.
                intFirst = InStr(1, rst1.Fields(1), ">") + 1
                intLen = InStr(6, rst1.Fields(1), "<") - intFirst
                strFname = Mid(rst1.Fields(1), intFirst, intLen)
' Move two records to process last name field.
                rst1.Move 2
            Else
' If the first name is blank, set a Boolean flag to skip the whole record.
                blSkip = True
            End If
' Process last name field.
            intFirst = InStr(1, rst1.Fields(1), ">") + 1
            intLen = InStr(6, rst1.Fields(1), "<") - intFirst
            strLname = Mid(rst1.Fields(1), intFirst, intLen)
' Process company name field.
            rst1.Move 2
            If rst1.Fields(1) <> "  <dd> </dd>" Then
                intFirst = InStr(1, rst1.Fields(1), ">") + 1
                intLen = InStr(6, rst1.Fields(1), "<") - intFirst
' If there is a leading blank in the company name field,
' see if you can find the name after the blank.
                    If InStr(2, rst1.Fields(1), " ") <> 0 Then
                        intLen = InStr(6, rst1.Fields(1), " ") - intFirst
                    End If
' The parsing rule for the company name field converts with the new
' VBA Replace function html's &quot; into a single apostrophy.
                strCname = Replace(Mid(rst1.Fields(1), intFirst, intLen), "&quot;", "'")
            Else
' Set company name to zero-length string if there is no entry for the field.
                strCname = ""
            End If
.
.
.
' If Boolean skip flag is False, copy converted contact information
' to rst2, which is reference for WebBasedList table.
            If blSkip = False Then
                With rst2
                    .AddNew
                        .Fields("FirstName") = strFname
                        .Fields("LastName") = strLname
                        .Fields("CompanyName") = strCname
                        .Fields("Address1") = strSt1
```

```
                .Fields("Address2") = strSt2
                .Fields("City") = strCity
                .Fields("StateOrProvince") = strSt
                .Fields("PostalCode") = strPostal
                .Fields("Country") = strCountry
            .Update
        End With
    End If
End If

' Move to next record in temp1 table and start search
' for a record including label for first name.
        rst1.MoveNext
    Loop

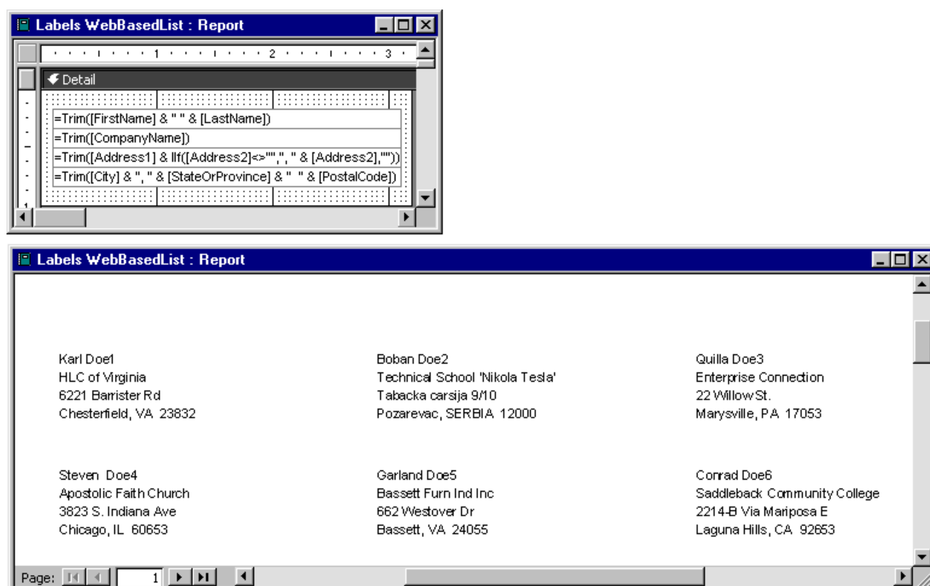
End Sub
```

כל אחד משלושת השדות המומרים מתבסס על לוגיקת ניתוח שונה במקצת. שאר השדות מטופלים בצורה יותר עקבית. יחד עם זאת, קיימים כמה הבדלים ברורים בחוקי הניתוח בין שדות חלופיים. באפשרותך לקרוא את ההערות ולחקור את הקוד כדי להבין טוב יותר פונקציות VBA שימושיות שמעבדות מחרוזות. בתום תהליך ההמרה, הקוד בודק ומוודא שלא היתה המרת שדה כלשהי שקבעה את דגל הדילוג בתור True. ערך False של הדגל מאפשר לשיגרה להוסיף רשומה חדשה לטבלת הקשר היחסית בתבנית המוצגת בתרשים 6.4. לאחר מכן השיגרה עוברת לרשומה חדשה בטבלת המקור (עותק ספר האורחים של FrontPage). כשטבלת המקור מחזירה את משתנה EOF בתור True, מסתיימת לולאת Do החיצונית והשיגרה נעצרת.

יצירת תוויות מען

תרשים 6.5 מציג את תצוגת העיצוב של דוח תוויות מען וגם קטע מהתצוגה **הצג לפני הדפסה** (Preview) של תוויות המען. התצוגות מבוססות על נתוני אנשי הקשר מהטבלה שבתרשים 6.4.

Access כולל אשף תוויות מען גרפיות שמתאימות למדבקות בשלל גדלים עבור מיגוון דגמי מדפסות לייזר ואחרות. האשף מאפשר גם להגדיר גדלים חדשים של תבנית תווית. הוא מבטל את הצורך לכתוב ביטויים לטיפול במחרוזות כדי ליצור את מבנה התוויות. החלון העליון בתרשים 6.5 מציג את שדות התווית בצורה מוגדלת מהרגיל שמאפשרת להבחין בביטויים שהם מכילים. האשף מתאים אוטומטית את גודל תיבות הטקסט המכילות את ביטויי המחרוזת שלו, כך שהתוויות מתאימות לטופס.



תרשים 6.5: החלון העליון מציג את דוח תוויות המען בתצוגת עיצוב; החלון התחתון מציג קטע מדף התוויות במצב תצוגה לפני הדפסה.

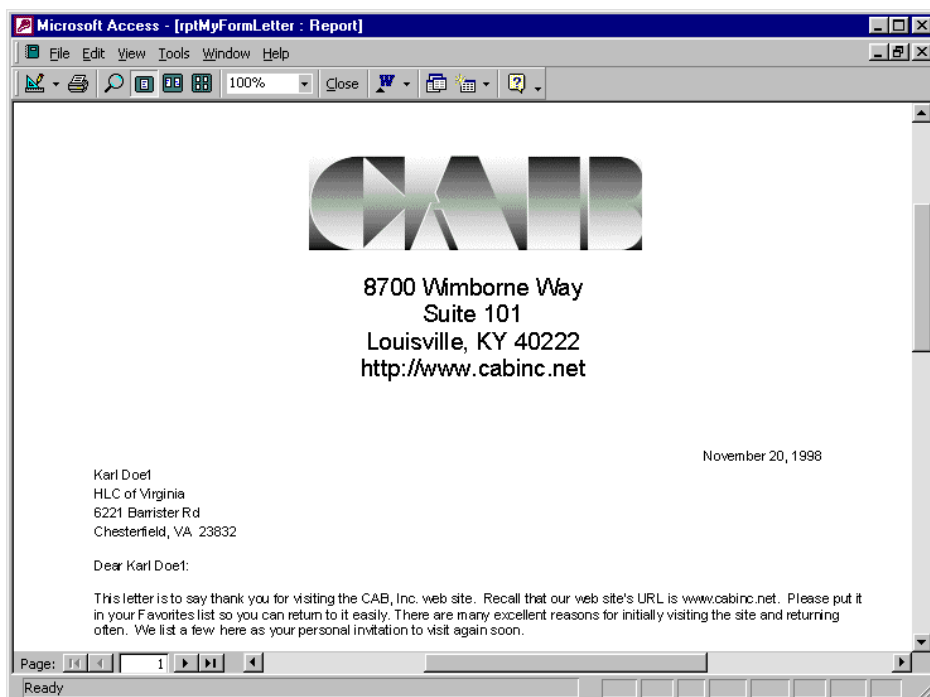
יצירת מכתב אחיד

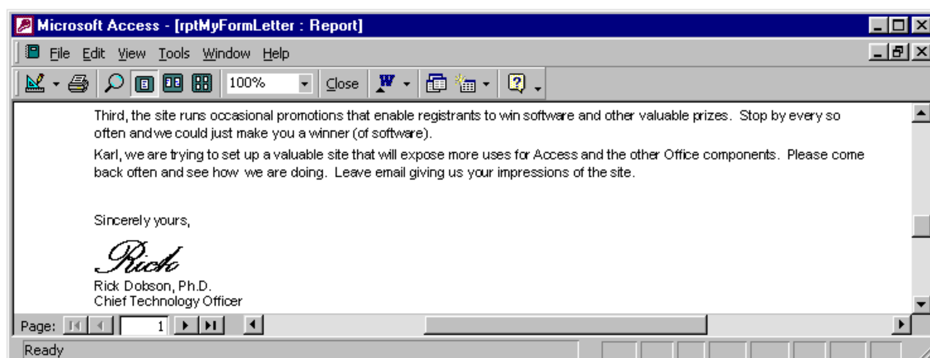
שימוש טיפוסי נוסף בטבלת אנשי קשר הוא במכתב אחיד. שילוב של פקדים, קבועים ומשתנים של מחרוזת ופונקציות VBA מותאמות אישית מאפשר ליצור מכתב אחיד, כגון זה שמוצג בתרשים 6.6. המכתב מפגין כמה תכונות ראויות לציון אופייניות לדוח של Access:

- בראש המכתב מופיע לוגו.
- פרטי מען לשיגור מענה מופיעים בגופן שונה מזה של גוף המכתב.
- תאריך המכתב מציג את שם החודש.
- כתובת הדואר היוצא והברכה משתנות מרשומה לרשומה.
- הפיסקה האחרונה במכתב משתנה אף היא לכל רשומה.
- דברי הסיום מלווים בחתימה.

המרת מספר חודש לשמו (כגון המרת הערך 1 ל"ינואר") לא היתה מעולם בעיה מיוחדת בתכנות, אך ב- Access 2000 וב- VBA 6 הדבר פשוט עוד יותר. תיבת הטקסט המציגה את התאריך מגלה את החידוש ודרך אפשרית לניצולו. היא מכילה את מחרוזת הביטוי הבאה:

```
=ThisMonthName() & " " & Day(Date()) &  ", " & Year(Date())
```





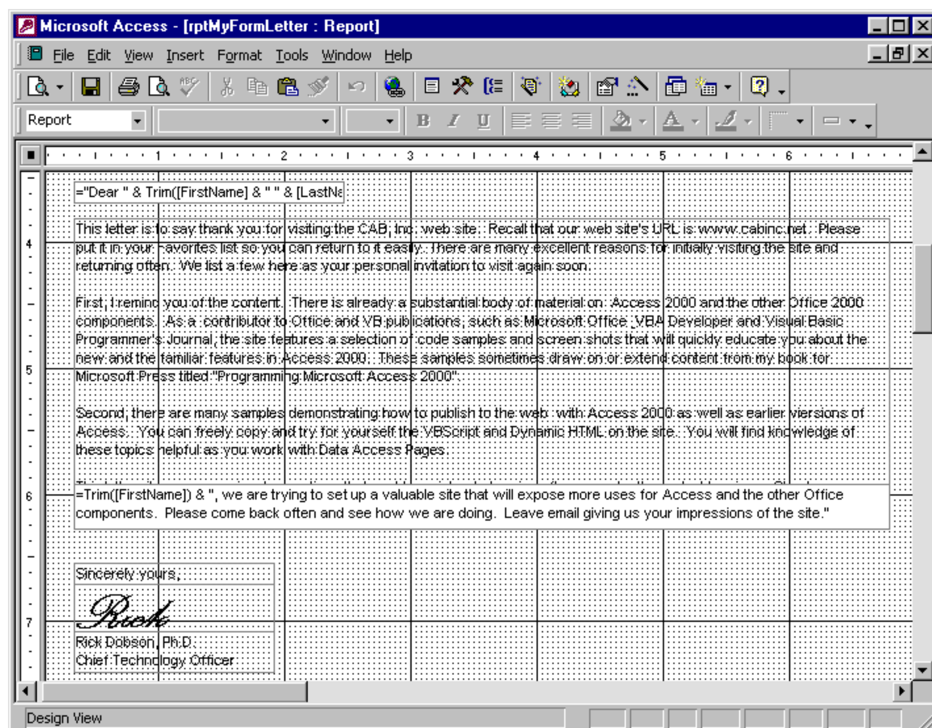
תרשים 6.6: קטע ממכתב אחיד

הביטוי מכיל זוג פונקציות מקוננות מוכללות עבור היום והשנה, אך פונקציה מותאמת אישית (ThisMonthName) מחזירה את שם החודש. לפניך פונקציה מותאמת אישית ב-VBA 6, ThisMonthName, שממירה את מספר החודש בשם המתאים. הפונקציה פוטרת אותך מהצורך לכתוב את המשפט Select Case בשגרת פונקציה, או להפעיל פונקציה מסוג Choose כדי לגרום להצגת שם חודש בדוח.

```
Public Function ThisMonthName()
   ThisMonthName = MonthName(Month(Date))
End Function
```

ייתכן שתהית מדוע לא ניתן ליישם את הפונקציה MonthName ישירות בתיבת הטקסט בדוח. לו עשית זאת, היית גורם לשגיאה מכיון ש-MonthName אינה פונקציה של Access. עליך לקרוא לה בשיגרה ולהחזיר את התוצאה לאובייקט של Access, כגון תיבת טקסט בדוח.

תרשים 6.7 מציג את תצוגת העיצוב של דוח המכתב האחיד המכיל שילוב של פקדי תווית ותיבת טקסט. אם גוף המכתב אינו מכיל רכיבי התאמה לכל רשומה בנפרד, ניתן להציג את כל הטקסט באמצעות תווית פשוטה. לעומת זאת, מאחר שהפיסקה האחרונה מתחילה בשם הקשר הראשון, הדוח זקוק לדרך לשחזור ערך השדה FirstName. קל לעשות זאת בעזרת ביטוי מחרוזת בתיבת טקסט. החלק התחתון משלב בין ערך השדה וקבוע המחרוזת. ניתן לראות כי יש לנסות להציב את תיבת הטקסט ביחס לפקד התווית שלפניו. כדי לשפר את זרימת הטקסט בתצוגה המקדימה, עליך לקיים חפיפה בין פקדים בתצוגת העיצוב. ייתכן שיהיה צורך בכמה ניסיונות עד קבלת עיצוב נאות.



תרשים 6.7: חלקו התחתון של המכתב מתרשים 6.6, והפעם בתצוגת עיצוב

שים לב לשימוש בגופן script בפקד תווית שנועד לדמות חתימה. זוהי אפשרות בת-קיימא למקרים בהם לא נדרשת חתימה בפועל – לדוגמה, לדואר בתפוצה גדולה, כגון מכתבים אחידים.

מיון, קיבוץ וחישוב

אחד הייתרונות הגדולים של תצוגת העיצוב הוא יכולתה לקבץ ולמיין רשומות להצגה בדוח. יכולת זו מושתת על שני גורמים: מקטעי פירוט של הדוח ותיבת הדו-שיח **מיון וקיבוץ** (Sorting And Grouping). אפשרות הקיבוץ מפשטת את חישוב סכומי הביניים לפי קבוצה וגם מחשבת את הסכומים הכוללים.

תרשים 6.8 מציג דוח פשוט שמדגים רכיבי מיון, קיבוץ וחישוב בסיסיים בדוח. הדוח בן שני העמודים מציג את כותרת הדוח בראש העמוד הראשון בלבד. פקד התוויות של כותרת הדוח ממוקם על-כן במקטע **כותרת עליונה של דוח** (Report Header). לעומת זאת, שתי כותרות העמודות מופיעות במקטע **כותרת עליונה של עמוד** (Page Header) מכיון שהן מופיעות בראש העמודות בשני עמודי הדוח. המקטע **פירוט** (Detail) מכיל זוג תיבות טקסט מאוגדות לשדות של מקור הרשומות של הדוח. שתי תיבות הטקסט בדוח מוצגות פעם אחת לכל רשומה במקור הרשומות שברקע. סדר המיון של שורות אלו יכול להיות בלתי תלוי בסדר הרשומות במקור הרשומות שברקע. יכולת זו משפרת את השימושיות הכוללת במקורות רשומות בעבודה עם דוחות רבים.

תרשים 6.8: דוח שמציג קיבוץ לפי תאריך

השורות שבתרשים 6.8 מקובצים לפי רבעון. תיבת הדו-שיח **מיון וקיבוץ** בתצוגת העיצוב משמשת לבחירת שדה אחד או יותר שלפיו מקבצים שורות. קיבוץ לפי שדה מוסיף לדוח את המקטעים כותרת עליונה וכותרת תחתונה של דוח, לכל קבוצה חדשה. יכולת הקיבוץ המוכללת מתאימה אוטומטית לסוג נתוני השדה שלפיו מקבצים את הדוח. במקרה של שדות תאריך, Access תומך בקיבוץ לפי שנה, רבעון, חודש, שבוע, יום, שעה ודקה. שדות מסוג **מספור אוטומטי** (AutoNumber), **מטבע** (Currency) או **מספר** (Number) מאפשרים לקבץ שורות לפי טווח מותאם אישית. באפשרותך לקבץ מחירי יחידת מוצר בהפרשים של $5 בדוח אחד, ובהפרשים של $10 בדוח אחר. באפשרותך גם לקבץ שדות **טקסט** (Text) בהתאם לתווים המובילים שלהם. הדבר מאפשר ליצור דוח בסגנון מילון שכל המוצרים מקובצים בו לפי שמם: ראשית מוצרים שמתחילים באות א׳, אחריהם מוצרים שמתחילים באות ב׳ וכן הלאה.

יכולות הקיבוץ תומכות בחישוב סכומי ביניים. שים לב שהדוח בתרשים 6.8 מסכם הזמנות מכירה לפי חודש ולפי רבעון. ניתן להפעיל פונקציית צבירה, כגון SUM, במקטע כותרת תחתונה של קבוצה כדי לחשב את פריטי הקבוצה. קיימות פונקציות צבירה נוספות שיכולות להוסיף ערך לדוח: AVG, COUNT, MAX ו-MIN. פונקציית הצבירה של סכום הביניים בתרשים 6.8 מופיעה בתיבת הטקסט במקטע **כותרת עליונה של Date** (Date Footer). ביטוי **מקור פקד** (Control Source) של תיבת הטקסט הוא

```
=Sum([CountOfOrderID])
```

ארגומנט הפונקציה מצביע על שדה ברקע של הפקד במקטע **פירוט**. שימוש ב-txtCountOfOrderID בתור ארגומנט שמצביע ישירות על פקד תיבת הטקסט גורם לשגיאה.



הערה:

כללי החישוב של סכומי ביניים נבדלים מאלה של שדות מחושבים אחרים בדוח. שדה מחושב רגיל, כגון כזה שבו מחשבים את הסכום לשורת פריט בהזמנה, מאפשר להפנות לערכי הפקד המסוימים ברשומה הנוכחית. הביטויים שתכתוב יכולים גם להפנות לשדות הרקע של פקדים. בעת חישוב סכומי ביניים, עליך להפנות לערך שדה הרקע.

יצירת דוחות מרובי עמודות

הדוגמאות שהוצגו עד כה בפרק אינן מציגות דוחות מסד נתונים קלאסיים בעלי עמודות רבות שמקורן בטבלת רקע אחת או יותר. באפשרותך ליצור דוח מסוג זה בשתי דרכים לפחות: באמצעות אשף הדוחות ובטכניקות עיצוב ידניות.

שימוש באשף הדוחות

אשף הדוחות (Reports Wizard), שהממשק שלו דומה לזה של אשף השאילתות הפשוטות, מאפשר ליצור דוח מרובה עמודות. תרשים 6.9 מציג דוח שהוכן באמצעות אשף הדוחות. כפי שניתן לראות, האשף מספק אמצעי חישוב רבים. מקור הרשומות שברקע הדוח מונה את מספר המכירות ומסכם את מחירי הסה"כ לפריט בחתך של לקוח. הדוח מסכם גם את כמויות המכירה והמחירים לפי לקוח ולכלל הלקוחות. לאחר מכן הוא מנצל את התוצאות שהתקבלו כדי לחשב את החלק היחסי מסך המכירות הכולל ואת מחיר הסה"כ ללקוח. הוא מסכם גם את מספר הרשומות לכל לקוח ומדפיס הודעה בת שורה אחת ובה שם הלקוח ומספר הרשומות שלו.

אשף הדוחות יכול להוסיף לדוח מיגוון מצודד של מאפייני עיצוב. תרשים 6.9 מציג אוסף בחירות אפשרי. השימוש בריווח ובקווים אופקיים מאפשר להבדיל את כותרת הדוח משאר חלקיו. באפשרותך גם לבחור את אפשרות פריסת השדות הרצויה מבין אפשרויות אחדות. פריסת הדוח שבתרשים ממקמת את משתנה הקיבוץ, CompanyName, בקצה השמאלי-קיצוני של עמודי הדוח. שים לב כי פריסה זו אינה דורשת כתיבת קוד כלשהו.

Microsoft Access - [Sales by Product within Customer]

File Edit View Tools Window Help

100% Close

Sales by Product within Customer

CompanyName	ProductName	CustomerID	Orders	ExtPrice
Alfreds Futterkiste				
	Aniseed Syrup	ALFKI	1	$60
	Chartreuse verte	ALFKI	1	$284
	Escargots de Bourgogne	ALFKI	1	$503
	Fløtemysost	ALFKI	1	$430
	Grandma's Boysenberry Spread	ALFKI	1	$380
	Lakkalikööri	ALFKI	1	$270
	Original Frankfurter grüne Soße	ALFKI	1	$21
	Raclette Courdavault	ALFKI	1	$825
	Rössle Sauerkraut	ALFKI	2	$604
	Spegesild	ALFKI	1	$18
	Vegie-spread	ALFKI	1	$878
Summary for 'CompanyName' = Alfreds Futterkiste (11 detail records)				
Sum			12	$4,273
Percent			0.56%	0.34%
Ana Trujillo Emparedados y he				
	Camembert Pierrot	ANATR	1	$340
	Gudbrandsdalsost	ANATR	1	$29

Page: 1

Ready

תרשים 6.9: קטע הדוח שלפניך מציג חלק מאפשרויות החישוב והעיצוב של אשף הדוחות

יצירת דוח מותאם אישית

כשיש צורך בסידור מיוחד של פקדים ושל חישובים ייחודיים בדוח, יש לעצב את הדוח בצורה ידנית. תרשים 6.10 מציג קטעים מדוח מותאם אישית שרכיביו התפקודיים דומים לאלה של הדוח שבתרשים 6.9. דוח מותאם אישית זה מעוצב בצורה פשוטה יותר כדי שיהיה קל יותר להבין את מרכיביו. החלון העליון מציג את המכירות לפי מוצר עבור לקוח יחד עם סכומי הכמויות של ההזמנה, מחיר סה״כ לפריט והאחוזים מסך כל ההזמנות עבור שני סוגי החישובים. החלון התחתון מציג את המשך הדוח, בו מתחיל הלקוח השני. מקטע הכותרת התחתונה של הדף מציג את תאריך הדוח, עמוד הדוח הנוכחי ומספר העמודים הכולל.

Microsoft Access - [qryCustomerProductSales]

File Edit View Tools Window Help

100% Close

Product Sales by Customer

ProductName	Orders	Extended Price
Sales for Alfreds Futterkiste		
Raclette Courdavault	1	$825
Aniseed Syrup	1	$60
Vegie-spread	1	$878
Rössle Sauerkraut	2	$604
Original Frankfurter grüne Soße	1	$21
Lakkalikööri	1	$270
Grandma's Boysenberry Spread	1	$380
Fløtemysost	1	$430
Escargots de Bourgogne	1	$503
Chartreuse verte	1	$284
Spegesild	1	$18
Alfreds Futterkiste Totals	12	$4,273
Percent of Grand Total	0.6%	0.3%

Page: 1

Ready

Microsoft Access - [qryCustomerProductSales]

File Edit View Tools Window Help

100% Close

Sales for Ana Trujillo Emparedados y helados		
Konbu	1	$60

This report prepared on: Saturday, November 21, 1998 — Page 1 of 103

Page: 1

Ready

תרשים 6.10: קטעים מדוח מותאם אישית מרובה עמודות

תרשים 6.11 מציג את תצוגת העיצוב של הדוח מתרשים 6.10. כמעט כל פקדי הדוח הם תוויות או תיבות טקסט. עליך להקצות קבועי מחרוזת לתוויות ולהגדיר את מאפיין הכיתוב שלהם בתור קבוע המחרוזת. הטקסט במקטע כותרת הדוח העליונה נמצא בתווית.

פקדים שמציגים שדות מספריים או ביטויי מחרוזת לעולם יהיו תיבות טקסט. ביטוי המחרוזת הראשון מופיע במקטע **כותרת עליונה של CompanyName**. הוא נוצר על ידי שרשור של קבוע מחרוזת והשדה CompanyName במקום הרשומות של הדוח. אחריו מופיע המקטע **פירוט**. כל אחת משלוש תיבות הטקסט שלו מפנה לשדה אחד. פקדים אלה אינם משלבים את השדה בתוך ביטוי, ולכן אין צורך לקדם אותם בסימון שוויון (=), או להציג את שם השדה בסוגריים מרובעים.

תרשים 6.11: תצוגת עיצוב של הדוח מתרשים 6.10

ארבעת הפקדים בצד ימין של המקטע **כותרת תחתונה של CompanyName** מכילים ביטויים לחישוב סכום הזמנות המכירות והסה״כ ללקוח, יחד עם חלקן היחסי של כל לקוח בסה״כ מספר היחידות והמחיר של הפריטים. הטבלה שלפניך מציגה את שמות הפקדים והביטויים שהם מכילים. ביטויי חישוב האחוזים מבוססים על שני

פקדים שמופיעים במקטעי הכותרת התחתונה של הדוח, txtGrandOrders ו-txtGrandExtPrice. פקדים אלה מחשבים את סה״כ כמות המכירות וסה״כ מחירי הפריטים של כל הלקוחות.

ביטויים לחישוב סכומים ואחוזים של תרשים 6.11

שם פקד	ביטוי
txtCustOrders	Sum([Orders])
txtCustExtPrice	Sum([ExtPrice])
txtOrdersPercent	=[txtCustOrders]/[txtGrandOrders]
txtExtPricePercent	=[txtCustExtPrice]/[txtGrandExtPrice]
txtGrandOrders	=Sum([Orders])
txtGrandExtPrice	=Sum([ExtPrice])

הסכומים של לקוחות יחידים והסכום הכולל מתייחסים ישירות לערכי שדות הרקע, Orders ו-ExtPrice. חישובי האחוזים של לקוחות בודדים מתייחסים לשמות הפקדים אשר תלויים בערכי השדות.



הערה:
כברירת מחדל, Access נותן לפקד תיבת טקסט שם הזהה לזה של השדה שאליו הוא מאוגד. כדי להימנע מבלבול, תוכל לתת לשדות שמות שונים. לדוגמה, אם שם השדה הוא Orders, באפשרותך לשנות את שם פקד תיבת הטקסט שמפנה אליו ל-txtOrders. הדבר יקל עליך לקרוא ביטויים ולקבוע אם הם מתייחסים לערך של שדה או לפקד.

שים לב בתרשים 6.10 כי תאריך הדוח מודפס בתחתית הדף, אך לפניו בא קבוע מחרוזת. אם ברצונך להחיל עליו תבנית, כגון התבנית Long Date (תאריך ארוך), עליך לבודד את הפונקציה Now בתיבת טקסט ללא רכיבים נוספים. תרשים 6.11 מציג שתי תיבות טקסט צמודות המכילות את מחרוזת הטקסט המובילה ואת הערך המעוצב של הפונקציה Now הבא אחריה.

צידו הימני של הדוח המוצג בתרשים 6.11 מנצל שתי מילות מפתח, Page ו-Pages, שמזהות את מספר דף הדוח הנוכחי ואת מספר הדפים הכולל בדוח (מילות המפתח אינן מופיעות בתרשים). שים לב כי באפשרותך לכלול מילות מפתח אלו בביטויי מחרוזת.

הפצת דוחות באמצעות תמונות

ניתן להפוך דוחות של Access לזמינים למשתמשים אחרים ב-WWW או בדואר אלקטרוני באמצעות קבצי תמונה (snapshot files). כדי שתחנת עבודה תוכל להציג תמונה של דוח של Access, יש להתקין בה מציג תמונות, אותו ניתן לקבל ללא תשלום בתור יישום נפרד, וגם פקד ActiveX לשימוש בדפדפן Web. תוכנת המציג מאפשרת לחוג משתמשים רחב לנצל את הדוחות שלך, כאלה ש-Access לא הותקן במחשבים שלהם, וגם כאלה שאינם מחוברים לרשת המקומית שלך. הדפדפן Netscape אינו תומך בפקדי ActiveX, אך יש באפשרותו לטעון קבצי תמונה דרך חיבור http. משתמשי Netscape יכולים להשתמש בגירסה העצמאית של מציג התמונות כדי להציג את קובץ התמונה השמור בכונן הקשיח שלהם.

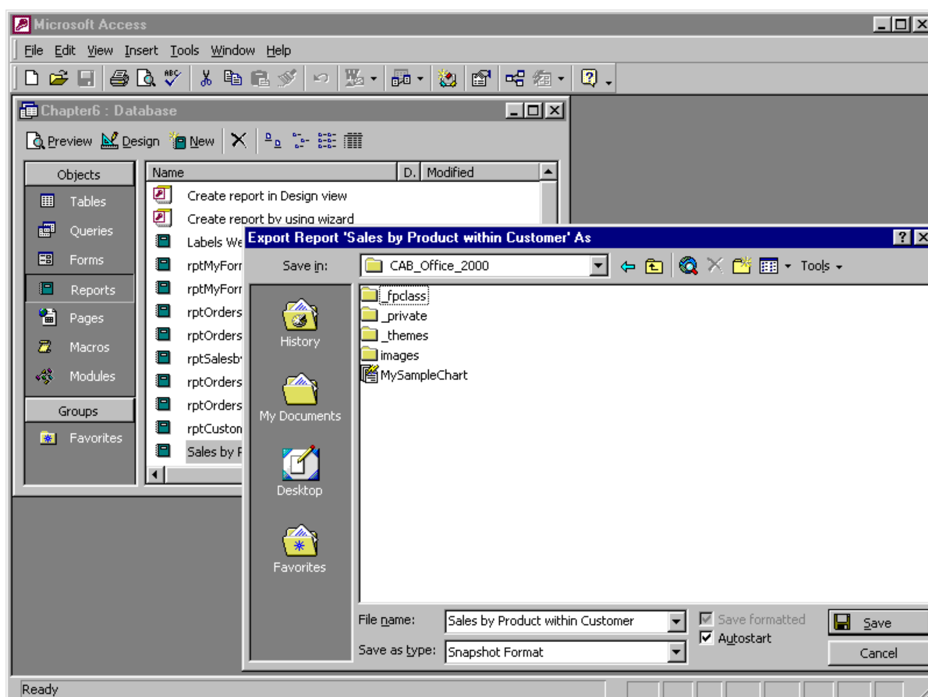
יצירת תמונה

באפשרותך ליצור תמונה (snapshot) של דוח על ידי בחירת הדוח בחלון מסד הנתונים ובחירה באפשרות **יצא** (Export) מתפריט **קובץ** (File). הדבר מכניס את שם הדוח בתור שם קובץ התמונה בתיבת הדו-שיח **יצא קובץ** (Export File) (תרשים 6.12). בתיבת הרשימה הנפתחת **שמור כסוג** (Save As), בחר **Snapshot Format**. בתיבת הרשימה הנפתחת **שמור ב** (Save In), הקצה מיקום עבור קובץ התמונה. תרשים 6.12 שומר את הדוח מתרשים 6.9 בספריה וירטואלית (שנקראת CAB_Office_2000) ברשת אינטרא-נט ארגונית. לחיצה על **שמור** (Save) בתיבת הדו-שיח שבתרשים 6.12 פותחת תיבת דו-שיח התקדמות ופותחת את הקובץ לאחר שמירתו, במציג התמונות (הכלול ב-Access). הקובץ מאוחסן בשרת, והתבנית שלו snp.*.

הצגת תמונה (Snapshot)

תחנת עבודה שמותקנים בה הדפדפן Internet Explorer מגירסה 3 ואילך ומציג תמונות (Snapshot Viewer), יכולה לפתוח קובץ תמונה. תרשים 6.13 מציג את הדוח ב-Internet Explorer 4. המציג נראה כאילו הוא חלק מהדפדפן, אך לאמיתו של דבר, הוא משתלט עליו. פקדי ניווט מיוחדים בשורה התחתונה של פקד ActiveX מאפשרים לנווט בין דפי הדוח. שורת הניווט כוללת גם לחצן הדפסה (מכיון שהפקד מבטל את רכיב ההדפסה המקומי של הדפדפן).

שים לב להתאמה הקרובה בין תמונת הדפדפן שבתרשים 6.13 לבין תמונת הדוח המקורי בתרשים 6.9. רמת התאמה זו אינה מתקיימת כשמייצאים את הדוח למסמך בתבנית HTML. בנוסף, ייצוא ל-HTML אינו מספק את תפקודיות הניווט המוכללת שמציע פקד מציג התמונות.



תרשים 6.12: יצירת קובץ תמונה באופן ידני

פקד מציג התמונות מאפשר להטביע דוח של Access בדף Web המכיל תוכן נוסף. קטע קוד HTML שלפניך מציג גוף דף Web ובו תגי H3 ו-H4 לפני הפניית האובייקט לפקד מציג התמונות. קטעי הטקסט הבאים לפני הפקד מסבירים כיצד לטעון את המציג אם הדוח אינו מופיע ומציגים היפר-קישור לאתר אינטרנט ממנו יכול המשתמש לטעון את הפקד. ההוראות מסבירות גם כיצד להציג את הדוח באמצעות Netscape Navigator. באופן טיפוסי תרצה לעדכן את הגדרת הערך Param, SnapshotPath, לפי כתובת ה-URL בה נמצא קובץ התמונה שברצונך להציג.

```
<body>
<H3>Snapshot Sample page</H3>
<H4>If you have an IE 3+ browser but cannot see the report below,
 download and install the
 <a href="http://www.microsoft.com/accessdev/prodinfo/snapdl.htm">
 Microsoft Access Snapshot viewer</a>. Then
 refresh the page. Netscape users will not even see the report
 container in their browsers, but they can open the report
 outside their browser using the same snapshot viewer mentioned
 above. Netscape users can download the snapshot file from
 a web server over http protocol to their workstation.</H4>
<OBJECT ID="SnapshotViewer" WIDTH=640 HEIGHT=480
CLASSID="CLSID:F0E42D60-368C-11D0-AD81-00A0C90DC8D9">
```

```
    <PARAM NAME="_ExtentX" VALUE="16722">
    <PARAM NAME="_ExtentY" VALUE="11774">
    <PARAM NAME="_Version" VALUE="65536">
    <PARAM NAME="SnapshotPath"
    VALUE="http://cab2200/cab_office_2000/MySampleChart.snp">
    <PARAM NAME="Zoom" VALUE="0">
    <PARAM NAME="AllowContextMenu" VALUE="-1">
    <PARAM NAME="ShowNavigationButtons" VALUE="-1">
</OBJECT>
</body>
```

http://cab2200/cab_office_2000/Sales%20by%20Product%20within%20Customer.snp - Microsoft Inte...

File Edit View Go Favorites Tools Help

Back Forward Stop Refresh Home Search Favorites History Channels Tools Fullscreen

Address http://cab2200/cab_office_2000/Sales%20by%20Product%20within%20Customer.snp

Sales by Product within Customer

CompanyName	ProductName	CustomerID	Orders	ExtPrice
Alfreds Futterkiste				
	Aniseed Syrup	ALFKI	1	$60
	Chartreuse verte	ALFKI	1	$284
	Escargots de Bourgogne	ALFKI	1	$503
	Fløtemysost	ALFKI	1	$430
	Grandma's Boysenberry Spread	ALFKI	1	$380
	Lakkalikööri	ALFKI	1	$270
	Original Frankfurter grüne Soße	ALFKI	1	$21
	Raclette Courdavault	ALFKI	1	$825
	Rössle Sauerkraut	ALFKI	2	$604
	Spegesild	ALFKI	1	$18
	Vegie-spread	ALFKI	1	$878
Summary for 'CompanyName' = Alfreds Futterkiste (11 detail records)				
Sum			12	$4,273
Percent			0.56%	0.34%

Opening page http://cab2 Local intranet zone

תרשים 6.13: הצגת הדוח בדפדפן Internet Explorer 4

שימושים נוספים בתמונות

קבצי תמונה בדוחות של Access משמשים למטרות נוספות לאלו שהוצגו. לדוגמה, ניתן לשגר תמונה בדואר אלקטרוני בתוך מסמך מצורף (attachment) להודעת דואר אלקטרוני על ידי לחיצה ימנית על הדוח בחלון **מסד נתונים** (Database) ובחירת

שלח אל (Send To) ולאחר מכן בחירה ב**נמען דואר** (Mail Recipient) מתפריט הקיצור. זכור לכלול בהודעת הדואר שלך קישור אל אתר האינטרנט ממנו ניתן להוריד את מציג התמונות, עבור נמענים שעדיין לא התקינו את המציג במחשב שלהם.

באפשרותך לנצל גם את השיטות DoCmd OutputTo ו-SendObject כדי למכן את ההמרה וההעתקה של קבצי תמונה לאתר שרת Web ברשת אינטרא-נט ארגונית או לנמעני דואר אלקטרוני. הסעיף האחרון בפרק כולל דוגמה שמנצלת קוד כדי לשגר קבצים רבים בדואר אלקטרוני אל נמען אחד או יותר. המשפט שלפניך מפרסם דוח מתוך מסד הנתונים Northwind באתר מקומי ברשת האינטרא-נט שלי:

```
DoCmd.OutputTo acOutputReport, "Alphabetical List of Products",
    "Snapshot Format", _
    "\\cab2200\c\inetpub\cab_office_2000\mysnapshot.snp", True
```

הפיכת דוח לדינמי

שלושה אירועי מקטע של דוח – Format, Retreat ו-Print – מאפשרים לבנות עיצוב ותוכן דינמיים לתוך דוח. Page, NoData, Close ו-Open הם אירועי דוח נוספים שיכולים לסייע בבניית דוחות נבונים. אירועים אלה יכולים גם לסייע לנהל את תפקוד היישום לפני פתיחת הדוח, במהלכה ואחריה. באפשרותך להשתמש בצירופים של אירועי דוח כדי ליצור עיצוב דוחות ואפקטים מיוחדים.

האירוע Open משמש להגדרת מאפייני דוחות ופקדיהם באמצעות קוד. זה האירוע הראשון של דוח. אם היישום יכול לקיים יותר מדוח פתוח אחד בו-זמנית, באפשרותך לנצל את האירועים Activate ו-Deactivate כדי לנטר את העברת המיקוד אל הדוח וממנו. השתמש באירוע Close כדי לבצע פעולות מיוחדות סמוך לסגירת דוח, כגון פתיחת טופס או הצגת תיבת הודעה.

האירוע NoData משמש לזיהוי דוח שמקור הרשומות שלו אינו מכיל נתונים כלשהם. אירוע זה מתרחש לאחר ש-Access מעצב דוח לצורך הדפסה. היישום יכול לנצל זאת כדי לבטל דוח העומד לפני הדפסה ואינו מכיל נתונים. באפשרותך גם לתכנת שגרות אירוע שמנחות את המשתמש לספק נתונים על ידי הזנת רשומות או באמצעות מעבר למקור רשומות אחר עבור הדוח.

עיצוב והוספת תוכן

דוגמאות הקוד שלפניך מעצבות ומוסיפות תוכן לדוח באופן דינמי באמצעות האירועים Print ו-Format. תרשים 6.14 מציג דוח שמנצל את האירוע Print בשלושה מקטעים כדי להוסיף מלבנים אדומים סביב מקטעי כותרת הדוח העליונה וכותרת העמוד התחתונה. שים לב להבדלי עובי הגבולות בין המלבנים. המקטע **פירוט** (Detail) מציג אליפסה סביב כל הסכומים הכוללים החודשיים הגדולים או שווים ל-30.

בעוד שהאירוע Print של כותרת הדוח העליונה קורה פעם אחת ויחידה בדוח, והאירוע Print של כותרת העמוד התחתונה קורה רק פעם אחת בדף, האירוע Print של המקטע **פירוט** מתרחש אחת לכל שורה בדף. משמעות הדבר שהדף שבתרשים 6.14 כולל 16 אירועי Print במקטע **פירוט**. כל אחד מהאירועים מאפשר ליישום לבדוק את ערכי הפקד עבור הרשומה הנוכחית. פירושו של דבר שבאפשרותך להציג בצורה סלקטיבית אליפסות סביב סכומי הזמנות חודשיות מסוימות.

Microsoft Access - [qryOrdersByMonth]

File Edit View Tools Window Help

77% Close

Orders by Month

Month and Year	Orders
07/01/1996	22
08/01/1996	25
09/01/1996	23
10/01/1996	26
11/01/1996	25
12/01/1996	31
01/01/1997	33
02/01/1997	29
03/01/1997	30
04/01/1997	31
05/01/1997	32
06/01/1997	30
07/01/1997	33
08/01/1997	33
09/01/1997	37
10/01/1997	38

This report prepared on:11/23/1998 12:12:42 AM Page 1 of 2

Page: 1

Ready

תרשים 6.14: האירוע Print בדוח זה מצייר מלבנים סביב מקטעי הכותרת העליונה של הדוח והכותרת התחתונה של העמוד. אירוע נוסף מצייר אליפסות בצורה סלקטיבית סביב סכומי הזמנות חודשיות בדוח

שלוש שגרות האירוע שלפניך הן הקוד שברקע הדוח של תרשים 6.14. היישום יכול להחיל את השיטה Line (כמו עבור שגרות האירוע של כותרת הדוח העליונה וכותרת העמוד התחתונה) כדי לצייר מלבן סביב מקטע הדוח. ארבעה משתנים מסוג Single מקבלים את הערכים עליון, שמאלי, רוחב וגובה של המקטע. משתנה מסוג Long מקבל את מספר הצבע של המלבן (אדום בדוגמה). רגע לפני הפעלת השיטה Line כדי לצייר את המלבן, השיגרה ReportHeader_Print מגדירה את רוחב הגבול כ-25 פיקסלים. שני זוגות קואורדינטות מציינות נקודות עבור השיטה Line. הערך שב-lngColor מציין את

צבע הגבול. הארגומנט האחרון של השיטה Line, ‏B, מורה לשיטה לצייר מלבן, או תיבה, תוך התייחסות לשתי הקואורדינטות בתור נקודות קצה הנמצאות בשני קדקודים מנוגדים.

```
Private Sub ReportHeader_Print(Cancel As Integer, PrintCount As Integer)
Dim sngTop As Single, sngLeft As Single
Dim sngWidth As Single, sngHeight As Single
Dim lngColor as Long
' Set top, left, width, & height.
    sngTop = Me.ScaleTop
    sngLeft = Me.ScaleLeft
    sngWidth = Me.ScaleWidth
    sngHeight = Me.ScaleHeight

' Set color.
    lngColor = RGB(255, 0, 0)
' Draw line as a box.
 Me.DrawWidth = 25
    Me.Line (sngTop, sngLeft)-(sngWidth, sngHeight), lngColor, B
End Sub

Private Sub PageFooterSection_Print(Cancel As Integer, PrintCount As Integer)
Dim sngTop As Single, sngLeft As Single
Dim sngWidth As Single, sngHeight As Single
Dim lngColor as Long

' Set top, left, width, & height.
    sngTop = Me.ScaleTop
    sngLeft = Me.ScaleLeft
    sngWidth = Me.ScaleWidth
    sngHeight = Me.ScaleHeight

' Set color.
    lngColor = RGB(255, 0, 0)
' Draw line as a box.
    Me.Line (sngTop, sngLeft)-(sngWidth, sngHeight), lngColor, B
End Sub

Private Sub Detail_Print(Cancel As Integer, PrintCount As Integer)
Dim sngHCtr As Single, sngVCtr As Single
Dim sngRadius As Single
' Position and size circle.
    sngHCtr = (Me.ScaleWidth / 2) - 3670
    sngVCtr = (Me.ScaleHeight / 2) - 20
    sngRadius = Me.ScaleHeight / 1.5
```

```
' Conditionally draw circle; last argument sets aspect ratio.
   If Me.CountOfOrderID.Value >= 30 Then
      Me.Circle (sngHCtr, sngVCtr), sngRadius, , , , 0.5
   End If
End Sub
```

ההבדל היחיד בין השגרות ReportHeader_Print ו-PageFooterSection_Print הוא השורה שמגדירה את רוחב גבול המלבן. המקטע **כותרת עליונה של דוח** (Report Header) מגדיר רוחב של 25 פיקסלים, אך המקטע **כותרת תחתונה של עמוד** (Page Footer) מצייר מלבן ברוחב ברירת מחדל של פיקסל אחד. שתי השגרות מציירות מלבן בשכבה שבקדמת שכבת הדוח הרגיל. ניתן לדעת זאת משום שהגבול האדום של השיטה Line מופיע על גבי צללית הרקע של כותרת הדוח הראשית.

שגרת האירוע של המקטע **פירוט** (Detail) מציירת את האליפסה סביב סכומי ההזמנות באמצעות השיטה Circle לכל שורה במקטע **פירוט**. עליך לקבוע באופן ניסיוני את המרכזים האנכי והאופקי וגם את רדיוס העיגול. ארגומנט יחס רוחב-גובה (aspect ratio) של השיטה Circle משמש להמרת עיגול לאליפסה ארוכה או צרה. בדומה לשיטה Line, השיטה Circle מציגה את הפלט שלה בשכבה שבקדמת שכבת הדוח הרגיל. הטבעת השיטה Circle במשפט If...Then מאפשרת לשיגרה לצייר בצורה מותנית את האליפסה סביב חלק מערכי השדה Orders, אך לא כולם.

סיכום ערכי דף

אם הדוח כולל דרישה להצגת סכום הערכים בדף הדוח, עליך לתכנת את סכומי הדף באמצעות שגרות אירוע, מכיון ש-Access אינו כולל אפשרויות מוכללות לביצוע משימה זו. הדוגמה של הספר למשימה זו מנצלת תיבת טקסט שהמאפיין **סכומים מצטברים** (Running Sum) שלה הוגדר בתור **מעל הכל** (Over All). הדבר גורם לתיבת הטקסט לסכם את השדה Control Source לאורך כל הדוח. תרשים 6.15 מציג את הדף הראשון והשני בדוח בעל עמודה נוספת לחישוב סכום מצטבר. העמודה הימנית-קיצונית גלויה למטרות למידה, אך בפועל מגדירים את המאפיין **גלוי** (Visible) של פקד הסכום המצטבר בתור **לא**.

באפשרותך לחשב סכומי דף בעזרת לא יותר משתי שגרות אירוע (שמוצגות בעמוד הבא). שגרת האירוע PageFooterSection_Format תופסת שתי שורות קוד בלבד. ראשית, היא מעתיקה את ערך הפקד בעל הסכום המצטבר (pagesum) אל lngCurrentRSum. לאחר מכן היא מגדירה תיבת טקסט נוספת במקטע כותרת הדף התחתונה (txtpagesum) בתור ההפרש בין lngCurrentRSum ו-lngLastRSum. הערך ההתחלתי של lngCurrentRSum הוא 0. בתום העיצוב של כל דף, אירוע הדוח Page מפעיל שיגרה שמעתיקה את הערך הנוכחי של lngCurrentRSum אל lngLastRSum, כך שההפרש lngCurrentRSum ו-lngLastRSum בשגרת האירוע Report_Page הוא סכום הדף של הדף הנוכחי.

Microsoft Access - [qryOrdersByMonth]

File Edit View Tools Window Help

77% Close

Orders by Month

Month and Year	Orders	
07/01/1996	22	22
08/01/1996	25	47
09/01/1996	23	70
10/01/1996	26	96
11/01/1996	25	121
12/01/1996	31	152
01/01/1997	33	185
02/01/1997	29	214
03/01/1997	30	244
04/01/1997	31	275
05/01/1997	32	307
06/01/1997	30	337
07/01/1997	33	370
08/01/1997	33	403
09/01/1997	37	440
10/01/1997	38	478

Orders on this page are: 478

This report prepared on:11/23/1998 1:07:48 AM

Page: 1

Ready NUM

Microsoft Access - [qryOrdersByMonth]

File Edit View Tools Window Help

77% Close

Month and Year	Orders	
11/01/1997	34	512
12/01/1997	48	560
01/01/1998	55	615
02/01/1998	54	669
03/01/1998	73	742
04/01/1998	74	816
05/01/1998	14	830

Orders on this page are: 352

This report prepared on:11/23/1998 1:09:02 AM

Page: 2

Ready

תרשים 6.15: הדוח מנצל שתי שגרות אירוע ותיבת טקסט במקטע **פירוט** שהמאפיין **סכומים מצטברים** שלה מוגדר בתור **מעל הכל**, כדי לחשב את סכום ההזמנות בדף

```
Public lngLastRSum As Long, lngCurrentRSum As Long
Public lngPageRSum As Long

Private Sub PageFooterSection_Format (Cancel As Integer, FormatCount As Integer)
   lngCurrentRSum = Me.pagesum
   Me.txtpagesum = lngCurrentRSum – lngLastRSum
End Sub

Private Sub Report_Page()
   lngLastRSum = lngCurrentRSum
End Sub
```

שים לב ששגרת האירוע PageFooterSection_Format שבדוגמה מחשבת ומציגה סכומי דפים על ידי כתיבת ערך לתיבת טקסט במקטע כותרת הדף התחתונה. שגרת האירוע Print אינה מאפשרת פעולה מסוג זה, מכיון שהאירוע Print מופעל רק לאחר שהדוח עבר את תהליך העיצוב. האירוע Format מופעל במהלך עיצוב הדוח באמצעות היישום.

עדכון דוח באופן דינמי

ניתן להשתמש ב-VBA כדי להוסיף מאפיין מקור רשומות חדש לדוח. בעת שהיישום משנה את תוכן הדוח, אפשר לעדכן את כותרת הדוח הראשית כך שזו תשקף את התוכן החדש. אם תציג את תוצאות השאילתה הפרמטרית בדוח, תוכל גם לעצב את התוצאות באמצעות אפשרויות עיצוב רבות.

תרשים 6.16 מציג טופס ודוח. המשתמש יכול לטפל בטופס כדי לשנות את תוכן הדוח. הטופס מכיל תיבת טקסט, קבוצת אפשרויות ובה חמש תיבות סימון ולחצן פקודה. לאחר הקלדת מספר בתיבת הטקסט ובחירת סוג השוואה, המשתמש יכול ללחוץ על לחצן הפקודה כדי לפתוח את הדוח שמימין בתצוגת עיצוב. תצוגה זו נחוצה כדי להוסיף מאפיין של מקור רשומות חדש וכדי להגדיר את המאפיין Caption של התוויות שמציג את כותרת הדוח הראשית. לאחר עדכון שני מאפייני הדוח באמצעות קוד, פותח היישום את התוצאות במצב **הצג לפני הדפסה** כדי להציג את תוצאות מקור הרשומות החדש. ההוראות הבאות לאחר הכותרת מסבירות כיצד לסגור את הטופס.

frmFormforProducts : Form
List Products Relative to UnitPrice
Amount: 55
Price comparison operator
Greater than
Greater than or equal to
Equal to
Less than or equal to
Less than
Print Products

Products
Products with a Unit Price <= $55
Close by close button (x); click Yes or No to complete action

ProductName	UnitPrice
Ipoh Coffee	$46
Tarte au sucre	$49
Manjimup Dried Apples	$53
Raclette Courdavault	$55

Page: 5

תרשים 6.16: הטופס שמשמאל מאפשר למשתמשים להקצות מקור רשומות וכותרת מתאימה לדוח שמימין.

שגרת האירוע cmdPrintThem_Click של לחצן הפקודה (בהמשך הסעיף) מבצעת שלוש משימות. ראשית, היא מרכיבה מחרוזת SQL בהתאם לבחירות שנעשו בטופס. השיגרה מוסיפה פסוקיות באופן סדרתי לחלקו הראשון של משפט ה-SQL שמציג את רשימת המוצרים ומחיריהם מתוך הטבלה Products. תחילה היא מוסיפה פסוקית WHERE למשפט הבסיסי בהתאם לתיבות הסימון שנבחרו ולכמות שהוקלדה בתיבת הטקסט. לאחר הוספת הפסוקית WHERE, השיגרה מוסיפה פסוקית ORDER BY שממיינת את קבוצת ההחזרה של מחרוזת ה-SQL, לפי מחיר היחידה. אם המשתמש בוחר באופרטור גדול מ (>) או באופרטור גדול או שווה ל (>=), השיגרה תקבע סדר מיון יורד. אחרת – קבוצת ההחזרה תמוין לפי מחיר היחידה בהתאם לברירת המחדל של סדר המיון.

הרכיב השני של השיגרה cmdPrintThem_Click מעדכן באמצעות קוד את מקור הרשומות ואת כיתוב התווית בדוח. לאחר פתיחת הדוח בתצוגת עיצוב, הרכיב מבצע משפט With...End With המבוסס על הדוח. כדי למנוע ערבוביה של מסכים, הרכיב השני מפעיל את השיטה Echo עם ארגומנט שערכו False. הדבר מעכב את ריענון המסך עד שהמשפט השני מפעיל את השיטה Echo עם ארגומנט שערכו True. בתוך קטע הקוד With...End With, השיגרה מגדירה את המאפיין RecordSource (מקור רשומה) של הדוח בתור מחרוזת SQL שנוצרה בחלק הראשון של השיגרה. לאחר מכן היא משנה את כיתוב התווית שמציגה את כותרת הדוח הראשית. משימה זו מתאפשרת הודות לביטוי מחרוזת שמציג את הערך מקבוצת האפשרויות ואת הכמות מתיבת טקסט.

הרכיב השלישי מבצע שני תפקידים: פותח את הדוח בתצוגת **הצג לפני הדפסה**, כך שהמשתמשים יכולים לראות אותו. לאחר מכן, הפקודה האחרונה משחזרת את הפונקציה Echo אשר מציגה את הדוח. כיבוי הפונקציה Echo והפעלתה מחדש מביאה למעבר מסכים חד וברור.

```
Private Sub cmdPrintThem_Click()
Dim strSQL As String, strOperator As String
Dim strWhere As String
' Set up SQL statement for report record source.
    strSQL = "Select ProductName, UnitPrice from Products"
    strOperator = Choose(optRule, ">", ">=", "=", "<=", "<")
    strWhere = "Where UnitPrice" & strOperator & txtAmount
    strSQL = strSQL & " " & strWhere & " Order By UnitPrice"
    If optRule <= 2 Then
        strSQL = strSQL & " Desc"
    End If
' The commented debug.print statement is convenient for debugging your SQL
statement; remove the comment when you change the SQL statement construction.
'     Debug.Print strSQL
' Open report in Design view to set the report's record source
and its label's caption.
    DoCmd.Echo False
    DoCmd.OpenReport "rptProductsfromForm", acViewDesign
    With Reports("rptProductsfromForm")
        .Visible = False
        .RecordSource = strSQL
        .Controls("lblTitle").Caption = _
        "Products with a Unit Price " & strOperator & " $" & txtAmount
    End With
' Now show the form to the user.
    DoCmd.OpenReport "rptProductsfromForm", acViewPreview
    DoCmd.Echo True
End Sub
```

טיפול בדוחות ובפקדי דוח באמצעות קוד

Access כולל שתי רמות הזנת נתונים בדוחות, בטפסים ובאובייקטים חשובים נוספים של מסד נתונים. בפרק 5 למדנו על האוסף AllForms. Access כולל אוספים נוספים: AllReports, AllTables, AllQueries, AllMacros, AllViews, AllModules, AllStoredProcedures, AllDataAccessPages ו-AllDataDiagrams. חבר בכל אחד מהאוספים הללו הוא אובייקט מסוג AccessObject (סוג אובייקט חדש ב- Access 2000). באפשרותך להפנות לחבר AllReports באחת הצורות הבאות:

```
AllReports (0)
```

```
AllReports ("name")
```

```
AllReports![name]
```

מספור דוחות

הקוד שתיצור יכול למספר אובייקטים מסוג AccessObject באוסף All כלשהו, כדי לקבוע אם האובייקטים נמצאים בחיבור של מסד נתונים. אין זה משנה אם האובייקט פתוח או סגור. באפשרותך גם לקבוע אם האובייקט טעון. כאשר אובייקט מסוג AccessObject טעון או פתוח, היישום יכול לעבוד עם מרחב אוסף מקבילי. אוספים כאלה הינם כולם רכיבים פתוחים, כגון דוחות וטפסים במסד נתונים של Access. חברי האוסף Reports הם דוחות בודדים פתוחים ביישום. אובייקטי Report פתוחים אלה חושפים את כל המאפיינים הזמינים באמצעות VBA במקום הקבוצה המוגבלת יותר באוסף AllReports. באפשרותך לנצל את המאפיין Name של AllReports ושל Reports כדי לעבור בין שני האוספים המקבילים. השימוש במאפיין IsLoaded באוסף AllReports מאפשר לוודא אם יש לפתוח דוח, טרם הניסיון לטפל במאפייניו ובשיטותיו.

השיגרה ListAllReports שלפניך ממספרת את חברי האוסף AllReports תוך הצגת שמם ומצב הטעינה שלהם. חברי האוסף AllReports שייכים ל-CurrentProject או ל-CodeProject, שהם חברים באובייקט Application. עליך להפנות לאחד מהם כדי לחשוף את חברי האוסף AllReports. לכן, השיגרה ListAllReports מתחילה בהגדרת הפניה אל החבר CurrentProject של האובייקט Application. תזדקק להפניה זו כדי לגשת לחברי האוסף AllReports. שים לב שהלולאה For...Each עוברת על כל אובייקט AccessObject (obj1) באוסף AllReports, אך הנתיב אל AllReports מתחיל בהפניה אל Application.CurrentProject.

```
Sub ListAllReports()
Dim obj1 As AccessObject, app1 As Object
' Create a reference to the current project instance.
    Set app1 = Application.CurrentProject
```

```
' List each report in the application and
' describe as loaded or not.
    For Each obj1 In app1.AllReports
        If obj1.IsLoaded = True Then
            Debug.Print obj1.Name & " is loaded."
        Else
            Debug.Print obj1.Name & " is not loaded."
        End If
    Next obj1
End Sub
```

האוספים AllReports ו-AllForms דומים זה לזה בתפקידם. אינך מוגבל לבדיקת חברים מסוג AccessObject בפרויקט הפעיל. השגרות ListAllFormsElsewhere ו-ListAllReportsElsewhere שלפניך מראות כיצד לתכנת את שני האוספים כשהם מצביעים על פרויקט אחר. שים לב לדמיון הקוד בין האוספים וגם לדמיון הקוד בין הפרויקט הנוכחי לפרויקט האחר.

השגרה ListAllFormsElsewhere שלפניך מדפיסה את מספר השמות הכולל של כל חברי האוסף AllForms שבקובץ Chapter5.mdb. הקובץ נמצא בתקליטור המצורף לספר. השיגרה מבוססת על ההנחה שהעתקת את הקובץ מהתקליטור אל התיקיה Programming Access. היא ממספרת את החברים (האובייקטים AccessObject) בקובץ מסד נתונים אחר.

```
Sub ListAllFormsElsewhere()
Dim appAccess1 As Access.Application
Dim obj1 As AccessObject
' Create a reference to another database file.
    Set appAccess1 = New Access.Application
    appAccess1.OpenCurrentDatabase "c:\Programming Access\" & _
        "Chap05\Chapter5.mdb"

' Print the total number of forms in the database.
    Debug.Print appAccess1.CurrentProject.AllForms.Count
    For Each obj1 In appAccess1.CurrentProject.AllForms
        Debug.Print obj1.Name
    Next obj1
End Sub
```

השיגרה ListAllReportsElsewhere שלפניך מעוצבת בצורה זהה לקודמתה, למרות שהיא מטפלת באוסף AllReports ולא באוסף AllForms ומנצלת את מסד הנתונים Northwind.mdb במקום את Chapter5.mdb. עיצוב השגרות כמעט זהה, למעט השימוש במשתני מחרוזת להגדרת שם מסד הנתונים. שינוי זה הוכנס משיקולי נוחות וכלליות גרידא – אין כלל ב-Access או ב-VBA שמכתיב שימוש במחרוזות.

```
Sub ListAllReportsElsewhere()
Dim appAccess1 As Access.Application
Dim obj1 As AccessObject
Dim srtPath As String, strFile As String, strDBName As String
' Create a reference to another database file.
    Set appAccess1 = New Access.Application
    strPath = "c:\Program Files\Microsoft Office\Office\Samples\"
    strFile = "Northwind.mdb"
    strDBName = strPath & strFile
    appAccess1.OpenCurrentDatabase strDBName

' Print the total number of reports in the database.
    Debug.Print appAccess1.CurrentProject.AllReports.Count
    For Each obj1 In appAccess1.CurrentProject.AllReports
        Debug.Print obj1.Name
    Next obj1
End Sub
```

שינוי מאפייני פקד דוח

קוד היישום שכתבת יכול לנצל את האוסף AllReports בתור נתיב לדוחות פתוחים ספציפיים ולפקדים שהם מכילים. לאחר שתעבור בנתיב זה, היישום יוכל לקרוא ולשנות את המאפיינים של דוחות פתוחים. השיגרה ControlsInReports ניגשת דרך חברי האוסף AllReports אל מאפייני תיבות הטקסט והתווית בדוחות פתוחים ספציפיים.

```
Sub ControlsInReports()
Dim obj1 As AccessObject, ctl1 As Control
    For Each obj1 In CurrentProject.AllReports
        If obj1.IsLoaded = True Then
            For Each ctl1 In Reports(obj1.Name)
                If ctl1.ControlType = 100 Then
                    Debug.Print ctl1.Name, ctl1.Caption
                ElseIf ctl1.ControlType = 109 Then
                    Debug.Print ctl1.Name, ctl1.Value
                Else
                    Debug.Print ctl1.Name & " is not a label or a text box."
                End If
            Next ctl1
        End If
    Next obj1
End Sub
```

השיגרה ControlsInReports פותחת בלולאת For...Each שעוברת על חברי האוסף AllReports. אם היא מזהה חבר פתוח (המאפיין IsLoaded שלו מכיל ערך True), הקוד נכנס ללולאת For...Each מקוננת כדי למספר את פקדי הדוח. באפשרותך לנצל את המאפיין ControlType כדי לקבוע את סוג הפקד. חשוב לזהות את סוג הפקד, מכיון שהסוג קובע את המאפיינים שחושף הפקד. לדוגמה, פקד תווית מציג את המאפיין Caption שלו, אך פקד תיבת טקסט מנצל את המאפיין Value כדי לתאר מה הוא מציג. באפשרותך לנצל את סורק האובייקטים (**Object Browser**) ב-VBE כדי להציג את הקודים המספריים של סוגי פקדים אחרים שברצונך לערוך או לבדוק.

משלוח תמונות

הדוגמה שלפניך ממספרת דוחות כדי לקבוע אם הם מסומנים לשיגור בדיוור בתור קבצי תמונה. הדוגמה מתבססת על שתי שגרות. השיגרה הראשונה, SendSnapShots ממספרת את חברי האוסף AllReports. מאחר שהקוד בודק אם המאפיין Tag (תג) של הדוח הוא "mail it", על הדוח להיות פתוח. המאפיין Tag אינו זמין באמצעות האוסף AllReports, אלא באמצעות האוסף Reports בלבד. השיגרה SendSnapShots בודקת את המצב IsLoaded של כל אחד מחברי האוסף AllReports. אם IsLoaded מכיל ערך False, השיגרה תפתח את הדוח לפני שתקרא לשיגרה השנייה. שגרת הדוגמה אינה קוראת לשיטה Echo עם הפרמטר False, ולכן המשתמש יכול לקבל בקלות משוב ברגע שהשיגרה השנייה נכנסת לפעולה. הדבר מתאים במיוחד לדוחות ארוכים בהם יצירה ושיגור של קובץ תמונה עלולים להימשך זמן רב.

```
Sub SendSnapShots()
Dim obj1 As AccessObject, app1 As Object
' Create a reference to the current project instance.
   Set app1 = Application.CurrentProject

' Enumerate each member in AllReports to verify if loaded.
' If not loaded, open before calling CheckMailItTag.
   For Each obj1 In app1.AllReports
      If obj1.IsLoaded = True Then
         CheckMailItTag obj1.Name
      Else
         DoCmd.OpenReport obj1.Name, acViewPreview
         CheckMailItTag obj1.Name
         DoCmd.Close acReport, obj1.Name, acSaveNo
      End If
   Next obj1
End Sub
```

```
Sub CheckMailItTag(obj1name)
Dim rep1 As Report
' Set reference to Reports member corresponding to AllReports member.
    Set rep1 = Reports(obj1name)
' If Tag property says "mail it" create a snapshot file and mail it.
    If rep1.Tag = "mail it" Then
        DoCmd.SendObject acOutputReport, obj1name, acFormatSNP, _
          "virginia@cabinc.net", , , "Snapshot Report", "Here is the report.", False
    End If
End Sub
```

השיגרה CheckMailItTag קולטת את שם הדוח שהועבר אליה באמצעות SendSnapShots. היא מנצלת שם זה כדי ליצור הפניה אל חבר האוסף Reports ששמו זהה לשם שהועבר אליה. לאחר מכן היא בודקת את המאפיין Tag של הדוח כדי לקבוע אם הערך שהוא מכיל הוא "mail it". אם כן, השיגרה מפעילה את השיטה SendObject של DoCmd כדי ליצור קובץ תמונה ולשלוח אותו לכתובת דואר אלקטרוני (virginia@cabinc.net, במקרה שלנו). באפשרותך להחליף את קבוע המחרוזת של הכתובת בכתובת כלשהי או בסדרת כתובות כפי שדורש היישום. חשוב שהארגומנט הבא לאחר גוף ההודעה יקבל ערך False. ערך ברירת המחדל True יגרום לשיגרה לעצור אם ההודעה פתוחה ולהמתין שהמשתמש יערוך את ההודעה. הגדרת ערך הארגומנט בתור False מאפשר לשיגרה לעבור בלולאה על כל הדוחות, ללא צורך כלשהו בהתערבות משתמש.

7

מודולי מחלקה, טופס ודוח

כדי לכתוב קוד ב- Microsoft Access בצורה יעילה, עליך לנהל את משאבי התכנות שלך כך שיהיה קל לעבוד איתם ולעשות בהם שימוש חוזר. ערך הקוד עולה ביחס ישר למה שניתן להפיק ממנו.

מודולי מחלקה מאחסנים קוד באופן שמאפשר לעשות בו שימוש חוזר. מודול המחלקה פועל כמו מכולה שחושפת את הקוד והמשתנים הנבחרים שבתוכה בצורה מוכרת למפתחים בסביבת Visual Basic for Applications (VBA). עקרונית, מפעילים שגרות מחלקה ומציבים ערכים במשתנים באמצעות תחביר זהה לזה המשמש לביצוע פעולות על מאפיינים ושיטות של אובייקטים מוכללים של Access. כדי לנצל את הקוד במודול מחלקה, אין צורך לדעת דבר על אופן פעולתו. כמו-כן, מאחר שמודולי מחלקה חושפים מאפיינים, שיטות ואירועים כפי שאובייקטים אחרים עושים זאת, גם מפתחים מתחילים בסביבת VBA יכולים להשתמש בהם.

בפרק זה נציג תחילה מודולי מחלקה עצמאיים וגם מודולי מחלקת טופס ודוח. לאחר מכן נדגים דרכים פשוטות לבניית מחלקות ביישומי Access ולפיתוח שיטות ומאפיינים מותאמים אישית. בהמשך נציג מקרה לדוגמה המשתמש בשלושה טפסים, בפונקציות מותאמות אישית אחדות מהסוגים Property Get ו- Property Let ובטכניקות המבוססות על אובייקטי נתונים ActiveX – ADO (ActiveX Data Objects) כדי להתחיל ליצור יישום. הסעיף שלאחר מקרה הדוגמה מציג את התחביר שבאמצעותו מתכנתים אירועים לתוך המחלקות המותאמות אישית ובו נפגוש במילת המפתח WithEvents.

בשלב הבא נתמקד במכולות של מודולי המחלקה, הטופס והדוח בעת שנבחן את אוספי All החדשים ב- Access 2000. פגשנו באוספים AllForms ו-AllReports; בפרק זה נכיר את האוסף AllModules (לאמיתו של דבר, קיימים בסך הכל עשרה אוספי All). הפרק מסיים בהסבר כיצד לשלב את האוסף AllModules עם האוסף Modules כדי לנהל את הקוד ביישום.

סוגי מודולים

קיימים שלושה סוגי מודולים:

- **מודולים סטנדרטיים** (Standard modules)**.** מודולים אלה מאחסנים שגרות משנה ושגרות פונקציה שברצונך להפוך לזמינות בכל חלקי קובץ מסד הנתונים. מודולים סטנדרטיים יכולים להכיל גם משתנים שהוגדרו באמצעות ההצהרה Public (ציבורי) שברצונך להפוך לזמינים עבור שגרות במודולים אחרים.
- **מודולי מחלקה עצמאיים** (Standalone class modules). מודולים אלה מאפשרים ליצור אובייקטים מותאמים אישית. תוכל להגדיר מאפיינים, שיטות ואירועים עבור אובייקטים אלה, ותוכל גם לנצל את מילת המפתח **New** כדי ליצור מופעים של אובייקטי הטופס.
- **מודולי מחלקה של טפסים ודוחות** (Class modules for forms and reports) (נקראים גם מודולי טופס ודוח). כברירת מחדל, טפסים ודוחות מבוססים על מודולים (הגדרת ברירת המחדל של המאפיין HasModule שלהם היא True). תוכל להשתמש במילת המפתח **Me** בהתייחסות למודול שנמצא מאחורי טופס או דוח.

מודולי מחלקה

מודולי מחלקה עצמאיים שונים ממודולי מחלקת טופס ודוח בכמה מובנים.

ראשית, בניגוד למודולי מחלקה של טופס ודוח, מודולי מחלקה עצמאיים אינם כוללים ממשק משתמש מוכלל. תכונה זו עושה אותם מתאימים יותר למשימות שאינן מחייבות ממשק, כגון חישובים, בדיקת מידע או ביצוע שינוי במסד נתונים. מודולי טופס או דוח שצריכים לבצע משימות עתירות חישוב, יכולים לקרוא למודולי מחלקה סטנדרטיים.

שנית, מודולי מחלקה עצמאיים כוללים את האירועים Initialize ו-Terminate שמאפשרים פעולות שצריכות להתבצע בעת פתיחה וסגירה של מופע של מחלקה. מודולי דוח וטופס אינם כוללים אירועים כאלה, אך באפשרותך לבצע דברים דומים באמצעות האירועים Load ו-Close.

שלישית, עליך להשתמש במילת המפתח New כדי ליצור מופעים של מודולים עצמאיים. מודולי מחלקת דוח וטופס מאפשרים ליצור מופעים גם באמצעות השיטות DoCmd OpenForm ו-OpenReport וגם על ידי הפניה אל המאפיינים או השיטות של מחלקת הדוח או מחלקת הטופס. לדוגמה, Form_MyForm.SetFocus פותחת את הטופס MyForm.

באפשרותך ליצור מודול מחלקה עצמאי מתוך התפריט **Insert** (הוספה) של VBE (תפריט זה כולל גם פקודות לבניית מודול או שיגרה סטנדרטיים). לאחר בניית מעטפת מודול מחלקה, ניתן לאכלס אותה בשגרות ובהצהרות ועל ידי כך לצייד אותה בשגרות ובמאפיינים מותאמים אישית.

פונקציות מאפיין ושיטות מותאמות אישית

פונקציות מאפיין מיוחדות מאפשרות לפתח בצורה קלה ופשוטה שילוב כלשהו של מאפיינים מסוג קריאה-בלבד, כתיבה-בלבד וקריאה/כתיבה עבור מחלקות. אם היישום שאתה כותב מרשה זאת, תוכל להגדיר מאפיינים באופן פשוט על ידי הגדרת משתנים ציבוריים. מאפיין שמוגדר על ידי מודול מחלקה באמצעות משתנה ציבורי, יהיה מסוג קריאה/כתיבה. היכולת להצהיר על מאפיינים מותאמים אישית מאפשרת להרחיב את מיגוון הרכיבים התפקודיים של Access לטפסים ודוחות. בנוסף, פונקציות מאפיין אלו מאפשרות ליצור מחלקות עצמאיות רבות עוצמה.

היישום יכול גם לבנות שיטות מותאמות אישית לתוך מחלקות. ניתן לנצל שגרות משנה או שגרות פונקציה כדי לבצע זאת. חשיפה סלקטיבית של משתנים ושגרות באמצעות מילת המפתח Public מאפשרת להגדיר בצורה ממוקדת את השיטות והמאפיינים שייחשפו. הדבר מאפשר ליישומים להגדיר ממשקים בינם לבין אובייקטי מחלקה שפועלים בדרכים מוגדרות ביותר.

יצירת מופעים של מחלקות

השיטות והשגרות הציבוריות תומכות בגישה באמצעות קוד של שגרות חיצוניות למחלקה. עליך ליצור תחילה מופע של המחלקה בשיגרה מארחת במודול אחר באמצעות מילת המפתח New (מילת מפתח זו משמשת גם ליצירת אובייקטים מתוך מחלקות אחרות, כגון האובייקטים Connection ו-Recordset של ADO. לאמיתו של דבר, היישומים שתיצור יכולים ליצור עותקים רבים של מחלקה מותאמת אישית בעת ובעונה אחת – בדיוק כמו המחלקות של ADO). לאחר יצירת מופע של מחלקה, קוד השיגרה המארחת מטפל במופע המחלקה, לא במחלקה עצמה. באפשרותך לשנות מאפיין של מופע אחד של טופס, אך כשתיצור מופע נוסף של הטופס, המאפיין שלו יקבל את ערך ברירת המחדל.

מחלקות ואירועים מותאמים אישית

למרות ש-VBA מאפשר ליצור מחלקות מותאמות אישית בעלות שיטות ומאפיינים נפרדים, לא ניתן לבנות אירועים מותאמים אישית בתוך מחלקות אלו. ניתן, לעומת זאת, לעצב מחלקה שמתחברת למחלקה מוכללת או ספריית סוגים שמצרפים אליה. לדוגמה, באפשרותך לבנות מודול מחלקה שמפעיל קוד VBA בתגובה לאירועים ItemAdded ו-ItemRemoved של האוסף References (הפניות). אוסף זה עוקב אחר קישורים אל ספריות סוג חיצוניות ופקדי ActiveX. לאחר הפעלת הפניה אל ספריה, כגון Microsoft ActiveX Data Objects 2.1, באפשרותך לבנות אירועים מותאמים אישית שמבוססים על אירועים של ADO עבור האובייקטים Connection ו-Recordset. אירועים אלה יכולים להעניק גישה אסינכרונית לנתונים, גישה שמאפשרת ליישום להגיב לפעולות המשתמשים, גם כשהיא מוכנה להגיב לחיבור מושלם או לזמינות של קבוצת ערכות רשומות שהובאה.

משתמשים במילת המפתח WithEvents בהצהרה Public כדי להצביע על הפניה לאובייקט שמנטר אירועים ומודיע עליהם מתוך פקד ActiveX. מילת מפתח זו חוקית במודולי מחלקה בלבד. באפשרותך להגדיר משתנים רבים במודול באמצעות מילת המפתח WithEvents, אך לא תוכל ליצור מערכים בעזרתה. בנוסף, הצהרה אינה יכולה לכלול בו-זמנית את מילות המפתח WithEvents ו-New.

שיטות ומאפיינים מותאמים אישית

בעת השימוש במודולי מחלקה, לא מן הנמנע שתעבוד עם שני מודולים נפרדים. מודול המחלקה חושף מאפיינים ושיטות וגם מפיץ אירועים. מודול שני מפנה אל מודול המחלקה; הוא מציב וקורא ערכים במאפיינים וגם מפעיל שיטות. מודול זה יכול ליזום פעולות שגורמות לאירועים, ואירועים אלה, יכולים להפעיל שגרות אירוע המשויכות להן במודול המחלקה.

חשיפת מאפיינים באמצעות משתנה ציבורי

הדוגמאות שלפניך מציגות שני תדפיסי קוד. האחד לקוח מהמודול MyTestClass. זהו מודול מחלקה שמתחיל בהצהרות על זוג משתנים – אחד עבור הדוגמה הנוכחית והאחר עבור הדוגמה הבאה. השיגרה שנקראת EP מחשבת את מחיר סה״כ לפריט מתוך שלושה ארגומנטים שמועברים אליה: units, price ו-discount. השיגרה שומרת את תוצאות הביטוי שלה במשתנה שנקרא ExtendedPrice. הצהרה באזור הכללי של המודול מגדירה את ExtendedPrice בתור משתנה ציבורי. הגדרתו כציבורי מאפשרת לשיגרה מארחת שנמצאת במודול אחר העובדת עם המופע של האובייקט MyTestClass, לקרוא את ערך המשתנה.

```
FROM MyTestClass module (a class module)
Public ExtendedPrice As Currency
Private MyComputedPrice As Currency
Public Sub EP(units As Long, price As Currency, discount As Single)
' Compute with result in public variable.
   ExtendedPrice = units * price * (1 - discount)
End Sub

FROM Module1 (a standard module)
Sub ComputeExtendedPrice()
' Create new instance of class module.
Dim MyPriceComputer As New MyTestClass
' Invoke EP method for class, and print Extended Price property.
   MyPriceComputer.EP 5, 5, 0.02
   Debug.Print MyPriceComputer.ExtendedPrice
End Sub
```

השיגרה המארחת, Compute ExtendedPrice, נמצאת במודול סטנדרטי שנקרא Module1. השיגרה יוצרת מופע של אובייקט, המבוסס על המחלקה שהוגדרה באמצעות האובייקט MyTestClass. בשלב הבא, היא מפעילה את השיטה EP של האובייקט. לבסוף, היא מדפיסה את המאפיין ExtendedPrice של האובייקט.

השיגרה שבדוגמה פשוטה ביותר, אך היא מדגימה מספר נקודות חשובות הקשורות לשימוש במודולי מחלקה:

- מודולי מחלקה הם בחירה הולמת לחישוב ביטויים קריטיים לעסקים. מודולי מחלקה משמשים בדרך כלל לאריזת פעולות מתוחכמות יותר מאלה הדרושות לחישוב סה"כ לפריט.
- השיגרה השנייה שנמצאת במודול הסטנדרטי, מתחילה בהפניה אל מודול המחלקה, MyTestClass. מילת המפתח New יוצרת מופע של אובייקט שמבוסס על המחלקה. בדוגמה שלנו, המשתנה שנקרא MyPriceComputer מפנה אל המחלקה.
- ניתן לנצל את הפניית האובייקט אל מופע המחלקה כדי להפעיל שיטות ולהגדיר או לקרוא ערכי מאפיין. מפנים אל השיטה EP של המחלקה באמצעות סימון הנקודה המקובל. מציגים את הארגומנטים אחרי שם השיטה ומפנים אל מאפיינים בעזרת כללי הסימון הבסיסיים.
- יצירת מאפיין של מחלקה יכולה להיות פעולה פשוטה בדיוק כמו הצהרה של משתנה ציבורי במודול המחלקה.

חשיפת מאפיינים באמצעות פונקציית מאפיין

הראשון בתדפיסי הקוד שלפניך מציג גישה שונה למשימה. הגישה מבוססת על מאפיין שהוגדר באמצעות הפונקציה Property Get. השיטה ep2 זהה כמעט בכל פרטיה לשיטה EP מהדוגמה הקודמת. ההבדל היחיד בין השתיים הוא ש-ep2 מפקידה את תוצאת הביטוי שלה במשתנה פרטי, ComputedPrice (עיין בהצהרת המשתנה הפרטי בדוגמה הקודמת). זה כשלעצמו אומר שמופעי המחלקה אינם חושפים את תוצאת הביטוי. הפונקציה Property Get היא זו שחושפת משתנה פרטי. מאחר שלא הוגדרה פונקציית מאפיין כלשהי עבור ComputedPrice, המאפיין הוא לקריאה בלבד. לו היתה פונקציה מסוג Property Let בשם זהה, המאפיין היה קריאה/כתיבה. השימוש במאפיינים מסוג קריאה בלבד יכול לסייע לאבטח את ערכי המאפיינים ביישום, או לפחות לספק דרכים להגדיר אותם.

```
FROM MyTestClass module (a class module)
Public Sub ep2(units As Long, price As Currency, discount As Single)
' Compute with result in private variable; expose
' result through Property Get function.
   MyComputedPrice = units * price * (1 - discount)
End Sub
```

```
Property Get ComputedPrice()
' This is how to return a read-only property.
   ComputedPrice = MyComputedPrice
End Property

FROM Module1 (a standard module)
Sub GetComputedPrice()
Dim MyPriceComputer As New MyTestClass
' Using a value defined by a property looks the same
' as one defined with a public variable.
   MyPriceComputer.ep2 5, 5, 0.02
   Debug.Print MyPriceComputer.ComputedPrice
End Sub
```

משתנים ציבוריים לעומת פונקציות מאפיין

תחביר הפעלת השיטה והדפסת ערך המאפיין זהים בשתי הדוגמאות, למרות השוני בחשיפת המאפיין. הדבר מאשש את הקביעה שמאפיינים מתנהגים בצורה זהה בין אם מגדירים אותם באמצעות הצהרה ציבורית, או בעזרת פונקציית מאפיין אחת או יותר. ייתכן שפשוט יותר ליישם מאפיינים במודולי מחלקה בעזרת משתנים ציבוריים, אך פונקציות מאפיין מצטיינות בגמישותן בחשיפת המאפיינים. השתמש בפונקציה Property Get לבדה לטיפול במשתנה מסוג קריאה בלבד, ובפונקציה Property Let לבדה לטיפול במשתנה מסוג כתיבה בלבד. לטיפול במאפיין מסוג קריאה/כתיבה, השתמש בשתי פונקציות המאפיין. אם המאפיין מפנה אל אובייקט ולא אל משתנה סקלרי, ניתן להשתמש בפונקציה Property Set במקום בפונקציה Property Let. הפונקציה Property Get משמשת להחזרת מאפייני אובייקט בין אם מטפלים במשתנה סקלרי או באובייקט.

מודולי מחלקה ומקורות נתונים

מודולי מחלקה מהווים אמצעי יעיל לאריזת קוד מסוג כלשהו. הם מכילים ערכים מיוחדים שעשויים להיות מועילים למקרים בהם יש צורך להפוך מקור נתונים לזמין לצורך עדכון או הצגה, אך בה בעת נדרש לאבטח את מקור הנתונים מפני נזק לא מכוון או רשלני.

עדכון נתונים באמצעות מחרוזת SQL

הדוגמה שלפניך מנצלת מודול מחלקה כדי לעדכן את השדה UnitsInStock בטבלה Products, בהתאם לשדה ProductID והכמות שהוזמנה. שיגרה בת שתי שורות מעבירה שני ארגומנטים לשגרת המשנה במודול המחלקה. דוגמה זו עושה שימוש במודול מחלקה שונה מאלה שהופיעו בדוגמאות החישוב של מחיר סה״כ לפריט

(MyTestClass2 במקום MyTestClass). בפועל, מחלקים את הפונקציות וההצהרות לאוספים אחידים של שגרות שיטה ומאפיינים שמייצגים מחלקות אובייקט נבדלות זו מזו. המשתנה OrderIt מייצג את המודול MyTestClass2. המודול מכיל פונקציה שנקראת PO1. הפונקציה קולטת שני ארגומנטים, אחד עבור ProductID ואחד נוסף עבור היחידות שהוזמנו.

```
Sub MyOrder()
Dim OrderIt As New MyTestClass
   OrderIt.PO1 1, 10
End Sub
```

השיגרה השנייה, PO1, מעדכנת את מסד הנתונים Products. במפורש, היא מפחיתה מהשדה UnitsInStock את מספר היחידות שהוזמנו. שיגרה זו נמצאת במודול המחלקה (MyTestClass2). שים לב לעיצוב השיגרה: היא מנצלת אובייקט מסוג Command עם מחרוזת SQL שמגדירה את שאילתת העדכון. למרות שהשיגרה קולטת שני ארגומנטים, היא אינה מיישמת שאילתת פרמטר. תחת זאת, היא מנצלת את הארגומנטים שהועברו בתור משתנים בביטוי המחרוזת שמגדיר את מחרוזת SQL. עיצוב זה מניב שיגרה מרוכזת שקל יחסית לקרוא אותה.

```
' A method for updating a table
Public Sub PO1(ProductID, units)
Dim cmd1 As Command
Dim strSQL As String
' Assign the command reference and connection.
   Set cmd1 = New ADODB.Command
   cmd1.ActiveConnection = CurrentProject.Connection
' Define the SQL string; notice
' the insertion of passed arguments.
   strSQL = "UPDATE Products " & _
      "SET UnitsInStock = " & _
      "UnitsInStock-" & units & " " & _
      "WHERE ProductID=" & ProductID
' Assign the SQL string to the command and run it.
   cmd1.CommandText = strSQL
   cmd1.CommandType = adCmdText
   cmd1.Execute
End Sub
```

עדכון נתונים באמצעות שאילתת פרמטר

מפתחים רבים מעדיפים לאמץ גישה מקובלת יותר שמתבססת על שאילתת פרמטר. השיגרה PO2 שלפניך מנצלת שאילתת פרמטר כדי לבצע את המשימה שביצעה PO1, והפעם באמצעות מחרוזת SQL. שאילתת פרמטר מאפשרת להצהיר על סוגי נתונים בעזרת כללי VBA מקובלים. שים לב שהקבוע adInteger של ADO מייצג את סוג

הנתונים long, והקבוע AddSmallInt מקצה את סוג הנתונים Integer. עליך ליצור את הפרמטרים באמצעות השיטה CreateParameter באותו סדר בו הצהרת עליהם בפסוקית Parameters במשפט השאילתה. אם לא תקפיד לעשות זאת, תקבל שגיאת זמן ריצה (RunTime Error).



הערה:

בדוק את המידע שמציגה העזרה המקוונת של Access על המאפיין Type של האובייקט Parameter של ADO, שם תמצא מיגוון סוגי נתונים עבור הצהרות של משתנים.

```
Public Sub PO2(ProductID As Long, units As Integer)
Dim cmd1 As Command
Dim strSQL As String
Dim prm1 As ADODB.Parameter, prm2 As ADODB.Parameter

' Assign the command reference and connection.
   Set cmd1 = New ADODB.Command
   cmd1.ActiveConnection = CurrentProject.Connection

' Write out SQL statement with parameters & assign to cmd1.
   strSQL = "Parameters PID Long,Quantity Integer;" & _
      "UPDATE Products " & _
      "SET UnitsInStock = " & _
      "UnitsInStock-Quantity " & _
      "WHERE ProductID=PID;"
   cmd1.CommandText = strSQL
   cmd1.CommandType = adCmdText
   ' Declare parameters; must have same order as declaration.
   Set prm1 = cmd1.CreateParameter("PID", adSmallInt, adParamInput)
   prm1.Value = ProductID
   cmd1.Parameters.Append prm1
   Set prm2 = cmd1.CreateParameter("Quantity", adInteger, adParamInput)
   prm2.Value = units
   cmd1.Parameters.Append prm2

' Run update query.
   cmd1.Execute

End Sub
```

משימת העדכון שמבצעת שאילתת הפרמטר מורכבת מארבעה שלבים עיקריים:

- הצהרה על משתנים והצבת הפניות.
- הגדרת מחרוזת SQL עבור שאילתת העדכון והצבתה במאפיין פקודה.
- יצירה והצבה של ערכים בפרמטרים שהוצהרו בשלב 2 לעיל.
- הפעלת הפקודה לעדכון מסד הנתונים.

מקרה לדוגמה: תכנות ממשק כניסה בסיסי

המקרה שלפניך מדגים גישה אפשרית לתכנות ממשק כניסה (login) בעזרת Access. השיגרה מתבססת על מודולי מחלקה עצמאיים וגם על מודולי מחלקת טופס. תהליך הכניסה ותוכן מודול המחלקה מנצלים טכניקות תכנות שמתאימות לכל משימה המחייבת שימוש בנתונים עם טפסים לא מאוגדים.

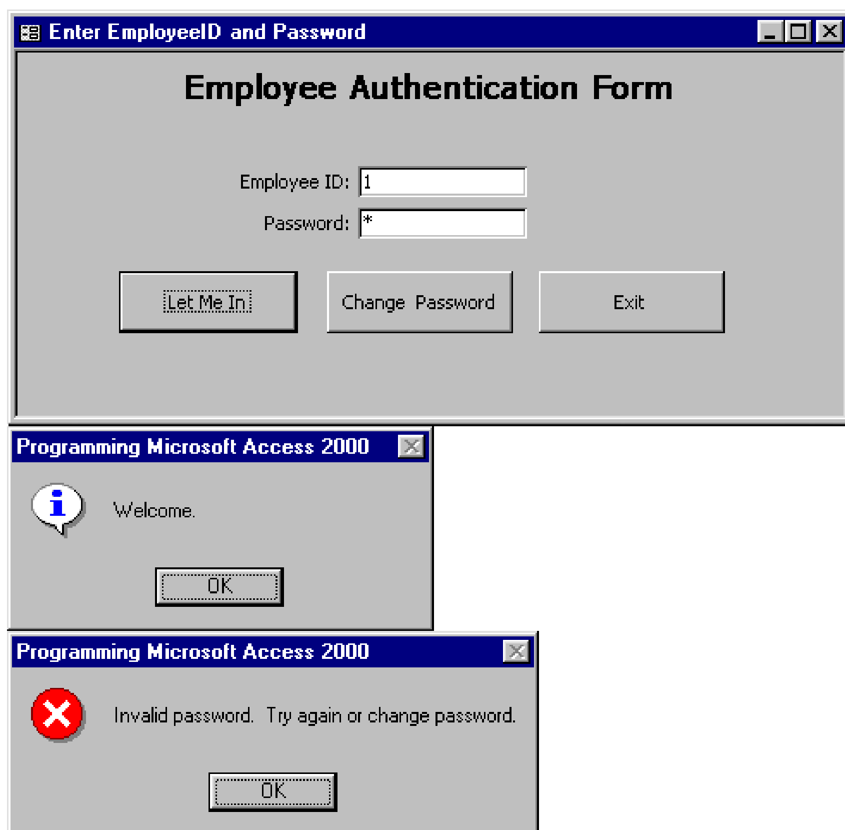
כדי להבליט את תפקידם של מודולי מחלקה ולשמור על שקיפות התהליך, שגרת הדוגמה אינה עושה שימוש ברכיב האבטחה המוכלל של Access. במקום זאת, היא מתבססת על זוג טבלאות ועל שלושה טפסים. הטבלה Passwords כוללת שני שדות בלבד: EmployeeID ו-Password. הטבלה Employees מיובאת ישירות ממסד הנתונים Northwind. השדה EmployeeID הוא המפתח הראשי שלה, והיא מכילה, בין היתר, נתונים עסקיים, אישיים ופרטי קשר של עובדים. שלושת הטפסים מתייחסים לתוכן הטבלאות כדי לנהל את תהליך הכניסה.



הערה:
מקרה הדוגמה מוותר על לכידת שגיאות, פעולה שאי אפשר בלעדיה במערכת מעשית. לכידת שגיאות היא מרכיב חיוני במיוחד במערכת שתלויה בקלט משתמש. לאחר שתכיר את המרכיבים הבסיסיים של מודולי מחלקה בפרק זה, כדאי שתרענן את ידיעותיך אודות לוגיקת לכידת שגיאות המתוארת בפרק 1.

טופס הכניסה הראשון

תרשים 7.1 מציג את הטופס הראשון יחד עם שתי תיבות ההודעה שביכולתו ליצור. תרחיש הכניסה נראה כך: המשתמש מקליד את שם המשתמש וסיסמתו בשתי תיבות הטקסט של הטופס ולוחץ על הלחצן **Let Me In**. אם הסיסמה מתאימה לערך השמור עבור השדה EmployeeID, היישום מציג תיבת הודעת פתיחה. אם אין התאמה בין הסיסמה השמורה לבין קוד זיהוי העובד, המשתמש יכול לנסות שנית, או להקליד ערך שונה.



תרשים 7.1: טופס הכניסה הראשון ובו שתי הודעות תגובה אפשריות

מסכת הקלט של הסיסמה

נוסף לקוד ברקע של הטופס ומודול המחלקה שהופעל באמצעות הטופס, עליך לבדוד בהקפדה את תיבת הטקסט **Password**. התיבה כוללת מסכת קלט של סיסמה שמציגה כוכבית כנגד כל תו שמקלידים בתיבה. מקצים את מסכת הקלט לתיבת הטקסט מתוך הכרטיסיה **נתונים** (Data) של תיבת הדו-שיח **מאפיינים** (Properties). לחץ על לחצן **בניה** (Build) הסמוך למאפיין **מסכת קלט** (Input Mask) כדי לפתוח תיבת דו-שיח שמאפשרת לבחור בה.

הקוד שברקע הטופס

המודול שברקע הטופס, המוצג להלן, מכיל שלוש שגרות אירוע – שיגרה אחת לכל לחצן. לחצן **Exit** סוגר את הטופס ותו-לא. לחצן **Change Password** פותח טופס נוסף ומעתיק אליו את ערך השדה מתוך הטופס הנוכחי. השיגרה שפותחת את הטופס frmWhoAmI גם מעבירה את המיקוד אל תיבת טקסט ריקה. לאחר מכן היא סוגרת את הטופס הנוכחי.

לחצן **Let Me In** מפעיל מודול מחלקה עצמאי (MyTestClass3). שים לב כי השיגרה מעבירה את תוכן שתי תיבות הטקסט שלה אל שגרת השיטה cpw במודול המחלקה. מודול זה מאתר את סיסמת העובד הקשורה לקוד הזיהוי שלו וקובע אם היא תואמת לסיסמה שבטופס. המחלקה מגיבה באחת משתי הודעות אפשריות (ראה תרשים 7.1). מודול המחלקה מפשט את הקוד של שגרת האירוע. כאן מתגלה יתרון נוסף של מודולי מחלקה: האפשרות לעבודת צוות. מפתחים מנוסים יכולים לכתוב שגרות בעלות דרגת מעורבות גבוהה במודולי מחלקה, בעוד שמפתחים פחות מנוסים יכולים לבצע משימות פיתוח צנועות יותר ולהפנות למודולי המחלקה כדי לשלב את קטעי הקוד שלהם באלה המתקדמים יותר.

```
Private Sub cmdExit_Click()

   DoCmd.Close

End Sub

Private Sub cmdNewPassword_Click()

   DoCmd.openform "frmWhoAmI"
   Forms("frmWhoAmI").txtEmpID = Me.txtEmpID
   Forms("frmWhoAmI").txtHireDate.SetFocus
   DoCmd.Close acForm, "frmInputPassword"

End Sub

Private Sub cmdLetMeIn_Click()

Dim PWT As New MyTestClass3
   PWT.cpw Me.txtEmpID, Me.txtPassword

End Sub
```

הפעלת מודול המחלקה

השיגרה cpw של מודול המחלקה MyTestClass3 מנצלת שאילתת פרמטר כדי לבדוק את הסיסמה של עובד מסוים בטבלה Passwords. אחד משני הארגומנטים המועברים לשיגרה הוא קוד העובד; השיגרה נותנת לפרמטר שלה את ערך הארגומנט. לאחר ביצוע האובייקט Command באמצעות שאילתת בחירה, השיגרה מציבה את קבוצת ההחזרה באובייקט Recordset. מאחר שהשדה EmployeeID הוא מפתח ראשי בטבלה Passwords, שאילתת הבחירה תחזיר תמיד רשומה בודדת.

השיגרה cpw מסתיימת בהשוואת הסיסמה שהחזירה השאילתה לסיסמה שהקליד המשתמש בטופס, בתור תנאי המשפט If...Then. אם יש התאמה, השיגרה מאפשרת למשתמש להיכנס ליישום. בפועל, פותחים טופס נוסף או אובייקט מסד נתונים אחר כלשהו שאליו מגבילים את הגישה באמצעות אבטחה מבוססת-סיסמה. אם אין התאמה, השיגרה מבקשת מהמשתמש להקליד את הסיסמה המדויקת מחדש.

```
Sub cpw(empid As Long, pw As String)
Dim cmd1 As Command
Dim strSQL As String
Dim prm1 As ADODB.Parameter
Dim rst1 As ADODB.Recordset
' Assign the command reference and connection.
    Set cmd1 = New ADODB.Command
    cmd1.ActiveConnection = CurrentProject.Connection
'Write out SQL statement with parameters & assign to cmd1.
    strSQL = "Parameters Secret Long;" & _
        "Select EmployeeID, Password from Passwords " & _
        "Where EmployeeID=Secret"
    cmd1.CommandText = strSQL
    cmd1.CommandType = adCmdText
    Set prm1 = cmd1.CreateParameter("Secret", adInteger, adParamInput)
    prm1.Value = empid
    cmd1.Parameters.Append prm1

' A handy line for catching SQL syntax errors
'    Debug.Print cmd1.CommandText

    cmd1.Execute

    Set rst1 = New ADODB.Recordset
    rst1.Open cmd1
    If rst1.Fields("Password") = pw Then
        MsgBox "Welcome on in.", vbInformation, _
            "Programming Microsoft Access 2000"
    Else
        MsgBox "Invalid password.  Try again or change password.", vbCritical, _
            "Programming Microsoft Access 2000"
    End If

End Sub
```

טופס הכניסה השני

תרשים 7.2 מציג את הטופס שמוצג בעת שהמשתמש בוחר לשנות את הסיסמה שהוקלדה עבור קוד העובד המתאים. כל תפקידו של הטופס מתמצה בהנחיית המשתמש לאשר את זהותו. המערכת דורשת את האישור בטרם תאפשר למשתמש לשנות את הסיסמה. בנסיבות רגילות, תיבת הטקסט הראשונה מתמלאת על ידי הטופס שטוען אותה (עיין בשיגרה הקודמת, cmdNewPassword_Click). כל שעל

המשתמש לעשות הוא להקליד את תאריך תחילת עבודתו וללחוץ על Submit. הדבר החשוב כאן, הוא לנצל שדה שערכו ידוע לעובד בלבד. באפשרותך לנצל שדה נוסף אחד או יותר כפי שתמצא לנחוץ.

frmWhoAmI : Form

Employee Confirmation Form

Employee ID: 1

Hire Date:

Submit

Exit

תרשים 7.2: טופס הכניסה השני שמנחה את המשתמש לאשר את זהותו

הקוד ברקע הטופס

הטופס מפעיל שאילתה כשהמשתמש לוחץ על לחצן Submit. מודול מחלקת טופס מעבד את נתוני השאילתה ומתאים את תוצאת קבוצת ההחזרה לקלט המשתמש. שגרת האירוע שברקע לחצן Submit כוללת משפט Dim שיוצר עותק של מודול המחלקה MyTestClass3 עם הפניה אל ProcessMe. שורת קוד שנייה מפעילה את השיטה WhoAmI של המחלקה, כפי שניתן לראות בקוד שלפניך.

```
Private Sub cmdSubmit_Click()
Dim ProcessMe As New MyTestClass3
   ProcessMe.WhoAmI CLng(txtEmpID), CDate(txtHireDate)
End Sub
```

הפעלת מודול מחלקת הטופס

שגרת בדיקת המידע של הטופס השני מופיעה בהמשך. היא מנצלת שאילתת פרמטר כדי לבצע בדיקה של תאריך תחילת ההעסקה של העובד בעל קוד הזיהוי הנתון. הקלדת המשתנים (שים לב לפונקציות CLng ו-CDate בשיגרה cmdSubmit_Click) לפני שעוברים למודול המחלקה, מאפשרת לנצל את יתרון אפשרות הקלדת הנתונים בהצהרה Parameters וגם את הקלדת הנתונים בטבלה. ללא הקלדה זו, היה על Access לבצע המרות פנימיות לסוג הנתונים Variant. העיצוב הבסיסי של הודעות הבדיקה וההחזרה דומה לזה של בדיקת הסיסמה. אם תאריך תחילת ההעסקה שבטופס מתאים לזה שבטבלה Employees, השיגרה תפתח את הטופס השלישי.

```
Sub WhoAmI(empid As Long, hd As Date)
Dim cmd1 As Command
Dim strSQL As String
Dim prm1 As ADODB.Parameter
Dim rst1 As ADODB.Recordset

' Assign the command reference and connection.
    Set cmd1 = New ADODB.Command
    cmd1.ActiveConnection = CurrentProject.Connection

' Write out SQL statement with parameters & assign to cmd1.
    strSQL = "Parameters InEID Long; Select EmployeeID, " & _
        "HireDate From Employees Where EmployeeID=InEID"
    cmd1.CommandText = strSQL
    cmd1.CommandType = adCmdText
    Set prm1 = cmd1.CreateParameter("InEID", adInteger, adParamInput)
    prm1.Value = empid
    cmd1.Parameters.Append prm1

' A handy line for catching SQL syntax errors
    Debug.Print cmd1.CommandText

' Execute command.
    cmd1.Execute
'
 Check Input vs. Table HireDate.
    Set rst1 = New ADODB.Recordset
    rst1.Open cmd1
    If rst1("HireDate") = hd Then
        DoCmd.openform "frmChangePassword"
        Forms("frmChangePassword").txtEmpID = Forms("frmWhoAmI").txtEmpID
        DoCmd.Close acForm, "frmWhoAmI"
    Else
        MsgBox "HireDate not valid for EmployeeID.  Try again or Quit.", _
          vbCritical, "Programming Microsoft Access 2000"
    End If

End Sub
```

טופס הכניסה השלישי

תרשים 7.3 מציג את הטופס האחרון שמופיע כשהמשתמש לוחץ על לחצן Submit בטופס השני, לאחר הקלדת תאריך תחילת ההעסקה המדויק שלו. הטופס כולל שלוש תיבות טקסט. אחת מיועדת לקוד העובד (תיבה זו מתמלאת מעצמה בנסיבות רגילות). תיבת הטקסט השנייה מיועדת לסיסמה החדשה, והתיבה השלישית מיועדת לאישור הסיסמה. אם אין התאמה בין תיבות הטקסט האלו, המערכת מציגה למשתמש הודעת

התרעה מתאימה. אם המשתמש לוחץ על לחצן Submit מבלי להקליד ערכים בכל שלוש התיבות, תופיע הודעת תזכורת נוספת. לבסוף, אם הטופס עונה על שתי הדרישות הללו, מודול המחלקה שאליו פנה הטופס מעדכן את הסיסמה עבור העובד בטבלה Passwords.

תרשים 7.3: הטופס השלישי, שמאפשר למשתמש לעדכן את הסיסמה

הקוד ברקע הטופס

המודול שברקע טופס זה הוא המעניין ביותר במקרה שלפנינו. המודול מבצע אימות נתונים, במקום להעביר אותם למודול המחלקה. גם שיגרה זו מפעילה מודול מחלקה עבור משפט SQL שמעדכן את סיסמת העובד.

ההפרדה בין אימות הנתונים ועדכון מסד הנתונים מציגה היבט נוסף של יישום מודולי מחלקה – ביצוע משימות רגישות באמצעות מודול מחלקה. כך ניתן לתקנן את המשימות ולהבטיח ביצוע הולם. מרכיבי יישום נוספים שאינם מחייבים תקנון הם מועמדים טיפוסיים להתאמה אישית על ידי מחלקות משתמשי הקצה.

הפונקציות Property Get ו- Property Let לוגיקת אימות נתונים מיוחדת זו מבוססת על זוג הפונקציות Property Get ו- Property Let. האירוע AfterUpdate של כל אחת משלוש תיבות הטקסט מפעיל את הפונקציה Property Let. פונקציה זו מעדכנת את ערך המשתנה AllFilled ל-True או False (אם כל התיבות התמלאו בנתונים חוקיים, ערך המשתנה הוא True ; אחרת, הערך False).

הפונקציה Property Get משקפת את מצב שלוש תיבות הטקסט בעזרת המאפיין filledCheck של הטופס. השיגרה cmdSubmit_Click בודקת ערך זה כדי לקבוע אם כל שלוש התיבות מסומנות. אם הערך הוא False, השיגרה מציגה הודעה שמזכירה למשתמש להשלים את הקלדת הערכים בכל התיבות. אחרת, שגרת אירוע הלחיצה בודקת אם יש התאמה בין תיבת הטקסט של הסיסמה וזו של אישור הסיסמה. אם לא, תוצג הודעה נוספת שתזכיר למשתמש להקליד סיסמה תואמת. לבסוף, לאחר שהמשתמש עבר את שני השלבים, השיגרה מפעילה את השיטה NewPS של המופע המקומי של המודול MyTestClass3.

```
Private AllFilled As Boolean

Private Sub txtConfirm_AfterUpdate()
   Me.filledCheck = txtConfirm
End Sub

Private Sub txtEmpID_AfterUpdate()
   Me.filledCheck = txtEmpID
End Sub

Private Sub txtPassword_AfterUpdate()
   Me.filledCheck = txtPassword
End Sub
```

```
Public Property Let filledCheck(vntNewValu)
    If (IsNull(txtEmpID) Or txtEmpID = "") Or _
        (IsNull(txtPassword) Or txtPassword = "") Or _
        (IsNull(txtConfirm) Or txtConfirm = "") Then
        AllFilled = False
    Else
        AllFilled = True
    End If
End Property

Public Property Get filledCheck()
    filledCheck = AllFilled
End Property

Private Sub cmdSubmit_Click()
Dim UpdatePW As New MyTestClass3
    If Me.filledCheck = False Then
        MsgBox "Please complete all entries before " & _
            "submitting your new password.", vbInformation, _
            "Programming Microsoft Access 2000"
    ElseIf txtPassword <> txtConfirm Then
        MsgBox "Password and Confirm Password do not " & _
            "match.  Re-enter one or both.", vbInformation, _
            "Programming Microsoft Access 2000"
    Else
        UpdatePW.NewPW txtEmpID, txtPassword
    End If
End Sub

Private Sub cmdLogin_Click()
    DoCmd.openform "frmInputPassword"
    Forms("frmInputPassword").txtEmpID = txtEmpID
    Forms("frmInputPassword").txtPassword = txtPassword
    DoCmd.Close acForm, "frmChangePassword"
End Sub

Private Sub cmdExit_Click()
    DoCmd.Close
End Sub
```

העברת ערכים לטופס אחר שתי שגרות נוספות משלימות את רכיביו התפקודיים של המודול שברקע הטופס השלישי. שגרת אירוע לחיצה ברקע של לחצן Employee Authentication מחזירה את המשתמש לטופס הראשון וממלאת את תיבות הטקסט של קוד העובד והסיסמה בערכים הלקוחים מהטופס השלישי. רכיב זה משחרר את

המשתמש מהצורך להקליד מחדש את הנתונים מייד לאחר האימות, אך החזרה לטופס הראשון מאפשרת נקודת כניסה יחידה ליישום. הדבר מפשט את תחזוקת היישום בטווח הארוך. לחצן היציאה של הטופס, Exit, סוגר את הטופס.

הפעלת מודול המחלקה

מודול המחלקה המופעל על ידי המודול שברקע הטופס השלישי, מנצל ביטוי מחרוזת כדי לחשב משפט SQL שאובייקט Command מעדכן בעזרתו את סיסמת העובד. זוהי דרך אפשרית לייצוג מחרוזת (כגון ערך הסיסמה) בתוך מחרוזת אחרת (משפט SQL השלם). שים לב לגרשים הכפולים הרבים, הן לפני והן אחרי ערך הסיסמה החדש. אלה קודי בקרה (escape codes) לייצוג גרש כפול בתוך זוג נוסף של גרשים כפולים. מלבד צורת הקינון של מחרוזת בתוך מחרוזת (בהתאם לכללי התחביר של VBA), הקוד קריא למדי. משפט קטע הודעה בסוף השיגרה מאשר את שינוי הסיסמה ומנחה את המשתמש כיצד להמשיך.

```
Sub NewPW(eid As Long, NuPassword As String)
Dim cmd1 As Command
Dim strSQL As String

' Assign the command reference and connection.
   Set cmd1 = New ADODB.Command
   cmd1.ActiveConnection = CurrentProject.Connection

' Define the SQL string; notice the insertion of passed arguments.
   strSQL = "UPDATE Passwords " & _
      "SET Passwords.Password = """ & NuPassword & """ " & _
      "WHERE EmployeeID=" & eid & ";"
   Debug.Print strSQL

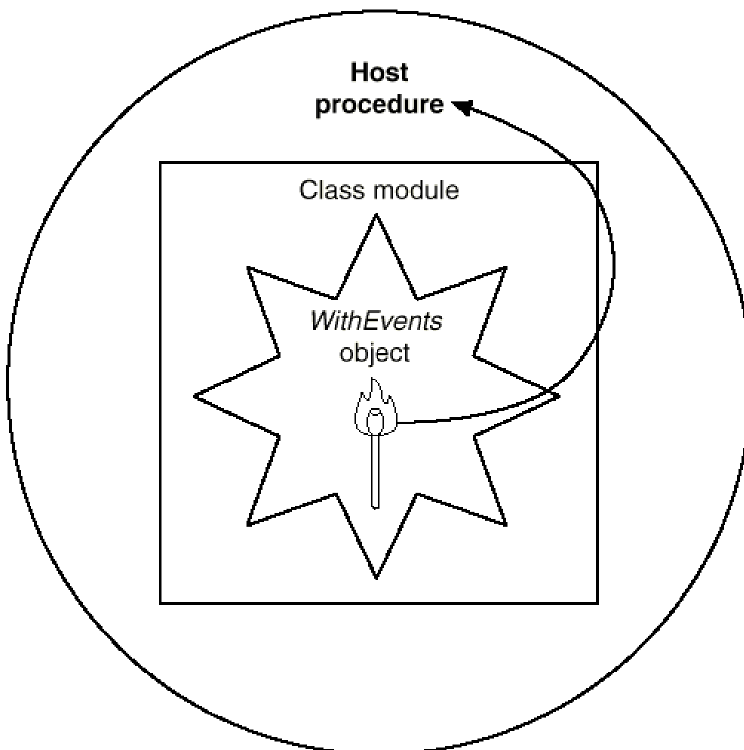
' Assign the SQL string to the command and run it.
   cmd1.CommandText = strSQL
   cmd1.CommandType = adCmdText
   cmd1.Execute

' Confirmation message
   MsgBox "Your new password is accepted.  " & _
      "Return to Employee Authentication or " & _
      "Exit this form.", vbInformation, "Programming Microsoft Access"
End Sub
```

תכנות אירועים לתוך מחלקות מותאמות אישית

באפשרותך לנצל את VBA ב-Access כדי ליצור מחלקות ליצירת מופעים של אובייקטים, אך אובייקטים של VBA אינם מסוגלים ליזום אירועים בעצמם. יחד עם זאת, תוכל ליצור מחלקות שמבוססות על ספריות סוג ופקדי ActiveX שמפיצים את האירועים שלהם בסביבה מארחת. במקרה של אובייקטים שמפיצים אירועים, יישום VBA יכול לארוז קוד סביב אירועים שמתרחשים בתוך המחלקה. כששיגרה מארחת של מופע מחלקה מפעילה שיטה שמחוללת הפצת אירוע מתוך המחלקה, שגרת האירוע מגיעה אל המארח של מודול המחלקה (ראה תרשים 7.4).

האוסף References מתייחס בצורה הירככית לאובייקט Application. מפעילים את השיטות AddFromFile ו-Remove כדי לאפשר ליישום להוסיף ולמחוק, באמצעות קוד, הפניות אל ספריות סוג. קבצי ספריה אלה מכילים תיאורים סטנדרטיים של אובייקטים, שיטות ואירועים חשופים. כזכור, מודל האובייקט ADODB תלוי בספריה. ניתן להוסיף הפניות אליו ולספריות נוספות בצורה ידנית או באמצעות קוד.



תרשים 7.4: מילת המפתח WithEvents מפיצה שגרות אירוע של אובייקט אל מארח האובייקט, בעת התרחשות של אירוע

פרק 2 מסביר כיצד להוסיף הפניה באופן ידני אל שלוש הספריות של ADO. הסעיף שלפניך מתאר כיצד להוסיף הפניות באמצעות קוד אל ספריה או פקד ActiveX כלשהם. באפשרותך ליצור הודעת אישור כשהיישום מסיים להוסיף או להסיר הפניה.

שתי מחלקות אירועים מוכללות

מודולי מחלקה כוללים שני אירועים מוכללים: Initialize ו-Terminate. האירוע Initialize מתרחש כשיוצרים מופע חדש של מודול מחלקה. יוצרים מעטפת עבור שגרת האירוע Initialize על ידי בחירת **Class** מתוך התיבה **Object** ו-**Initialize** מתוך התיבה **Procedure**. באפשרותך לעשות את כל הדרוש בשגרת האירוע Initialize כדי להכין את מופע המחלקה לשימוש.

שגרת האירוע Terminate היא כלי הניקוי לאחר ביצוע היישום הנוכחי. פעולת הניקוי יכולה להסתכם בהגדרת הפניית האובייקט בתור Nothing. האירועים Initialize ו-Terminate מתרחשים פעם אחת בלבד בתחילת חיי מופע המחלקה ובסיומם. זו הסיבה שאין הם שימושיים ביותר ליצירת התנהגות אינטראקטיבית או דינמית בהזדמנות כלשהי, למעט ביצירה ובפירוק של מופע של מחלקה.

שימוש במילת המפתח WithEvents ללכידת אירועים שהופצו

מודול המחלקה שלפניך מנצל את מילת המפתח WithEvents כדי ללכוד אירועים שהופצו באמצעות האוסף References. אוסף זה כולל פריט נפרד לכל פריט שנבחר בתיבת הדו-שיח References. האירועים ItemAdded ו-ItemRemoved מתרחשים רק כשהקוד מוסיף או מסיר הפניות. אם משתמש משנה את האוסף References באופן ידני, אירועים אלה אינם מופעלים.

```
Option Compare Database
' Declare object variable to represent References collection.
Public WithEvents evtReferences As References

' When instance of class is created, initialize evtReferences variable.
Private Sub Class_Initialize()
    Set evtReferences = Application.References
End Sub

' When instance is removed, set evtReferences to Nothing.
Private Sub Class_Terminate()
    Set evtReferences = Nothing
End Sub
```

```
' Display message when reference is added.
Private Sub evtReferences_ItemAdded(ByVal Reference As Access.Reference)
    MsgBox "Reference to " & Reference.Name & " added.", _
        vbInformation, "Programming Microsoft Access 2000"
End Sub

' Display message when reference is removed.
Private Sub evtReferences_ItemRemoved(ByVal Reference As Access.Reference)
    MsgBox "Reference to " & Reference.Name & " removed.", _
        vbInformation, "Programming Microsoft Access 2000"
End Sub
```

הפעלה וסיום של הפניה מסוג WithEvents

משתמשים במילת המפתח WithEvents בצירוף מחלקה שמפיצה אירועים. המשפט Public במודול המחלקה שהוצג קודם לכן, מצהיר על הפניה (evtReferences) אל האוסף References באובייקט היישום של Access. מילת המפתח WithEvents שבמשפט מאפשרת למודול המחלקה ללכוד אירועים שהופצו על ידי האוסף References. שגרת האירוע Class_Initialize מגדירה הפניה. כזכור, לא ניתן לנצל את מילת המפתח New עבור הפניה שהוצהרה באמצעות WithEvents.

אריזת קוד סביב אירועים שנלכדו

שתי שגרות אירוע במודול המחלקה, ItemAdd ו-ItemRemove, מפעילות משפטי קטע הודעה. משפטים אלה מציגים הודעות שנותנות שם להפניה ששיטה מוסיפה או מסירה. שגרות האירוע מציגות את תחביר הקוד המותאם אישית לאריזת אובייקטים שמפיצים אירועים. במקרה זה, האובייקט הוא האוסף References. שגרות האירוע רק כותבות את שם הספריה שהוספה או הוסרה מהאוסף References.

מודולים סטנדרטיים הגורמים לאירועים

בדומה למודול מחלקה כלשהו, יש צורך בשיגרה אחת או יותר של המודול הסטנדרטי כדי ליצור מופע של המחלקה (ראה דוגמה בהמשך) וכדי להפעיל שיטות, להציב ערכי מאפיין או לקרוא ערכי מאפיין. באזור ההצהרות של המודול המארח את המחלקה, כוללים את המשפטים Dim ו-Public עם מילת המפתח New ושם המחלקה. על ידי כך יוצרים מופע של המחלקה ומגדירים הפניה לאובייקט (objRefEvents בדוגמה).

תחביר מודול סטנדרטי של אירועים

אם המופע של המחלקה מפיץ אירועים מתוך אובייקט מוטבע, עליך להשתמש במשפט Public עם מילת המפתח WithEvents. משפט זה חושף את האירועים למודולים נוספים שמפנים למחלקה. כשמפעילים את השיטות מתוך מחלקת הרקע, יש להפוך את

הפניית האובייקט המקומית (objRefEvents), ההפניה שבמודול המחלקה החושפת את האירועים (evtReferences) ולאחר מכן שם שיטה ספציפית, כגון AddFromFile או Remove. בניגוד להפניה מקובלת למודול מחלקה, ההפניה הנוכחית מצביעה על שיטה של אובייקט המקור בהצהרה WithEvents.

```
' Create new instance of RefEvents class.
Dim objRefEvents As New RefEvents

Sub InvokeAddReference()
' Pass file name and path of type library to AddReference procedure.
   AddReference "C:\Program Files\Common Files\System\ado\msjro.dll"
End Sub

Sub InvokeRemoveReference()
' Pass name of existing reference. (Use internal name from File
' Properties list; same name appears when adding reference.)
   RemoveReference "JRO"
End Sub

Sub AddReference(strFileName As String)
   objRefEvents.evtReferences.AddFromFile (strFileName)
End Sub

Sub RemoveReference(strRefName As String)
   objRefEvents.evtReferences.Remove objRefEvents.evtReferences(strRefName)
End Sub
```

הדוגמה שלעיל מוסיפה הפניה לספריה שמאחסנת את המודל JRO (מודל זה מאפשר לבצע שכפול Jet באמצעות ADO). מפעילים את השיגרה InvokeAddReference כדי ליצור את ההפניה. השיגרה קוראת לשיגרה נוספת, AddReference, בעזרת תחביר להוספת פריט לאוסף References באמצעות מודול המחלקה RefEvents. הספריה המאחסנת את מודל JRO היא ספריית קישור דינמי (DLL) שנמצאת בדרך כלל בתיקיה ADO שהנתיב שלה הוא Program Files\Common Files\System\ADO הפעלת השיגרה InvokeRemoveReference מבטלת את הפריט JRO מתוך האוסף References. היעד של JRO הוא המאפיין Name של הפריט באוסף References.



הערה:

כדי לגלות את הארגומנטים של השיטות AddFromFile ו-Remove, הוסף הפניות באופן ידני. לאחר מכן, מספר את פריטי האוסף References באמצעות המשפט For...Each תוך כדי הצגת המאפיינים שלהם, Name ו-FullPath. השתמש בערכי מאפיינים אלה כדי לזהות בצורה ייחודית ארגומנטים של השיטות AddFromFile ו-Remove.

הרחבת היישום

באפשרותך לאמץ בקלות את המחלקה RefClass על ידי שימוש בשיטות AddReference ו-RemoveReference כדי לאפשר תהליך בחירה מקיף וגמיש יותר. לדוגמה, היישום שיצרת יכול לגזור את קלט השיגרה AddReference מתוך אוסף של ספריות סוג, קבצי הפעלה, פקדי ActiveX ואפילו מתוך קבצי מסד נתונים. ניתן להציג למשתמש תיבה משולבת ובה רשימת הפניות להוספה או להסרה. לחילופין, השיגרה יכולה לבחור מתוך רשימה המבוססת על גורמים נוספים, כגון יעדי המשימה של המשתמש.

שימוש באוספי ALL

אם אתה נמנה עם המפתחים הנוהגים לעקוב אחר האובייקטים שלהם בפרויקט מסד נתונים (רובנו רואים זאת כחיוני), תשמח לדעת על האוסף AllModules, מכר קרוב של האוספים AllForms ו-AllReports אותם פגשנו בפרקים 5 ו-6. חברי האוספים שמתחילים ב-All אינם אובייקטי מסד נתונים, כגון טפסים, דוחות ומודולים, אלא אובייקטים מהסוג AccessObject המכילים כמות מינימלית של פרטים אודות רוב הסוגים או אובייקטים שמורים במסד נתונים.

מאפייני AccessObject

ניתן למספר את אובייקטי AccessObject במהירות באוסף כלשהו. מאחר שאובייקטים אלה מצביעים על אובייקטים שמורים, לא ניתן להוסיף או למחוק חברים. מבצעים משימות אלו באמצעות האוספים הפתוחים שאליהם הם מצביעים.

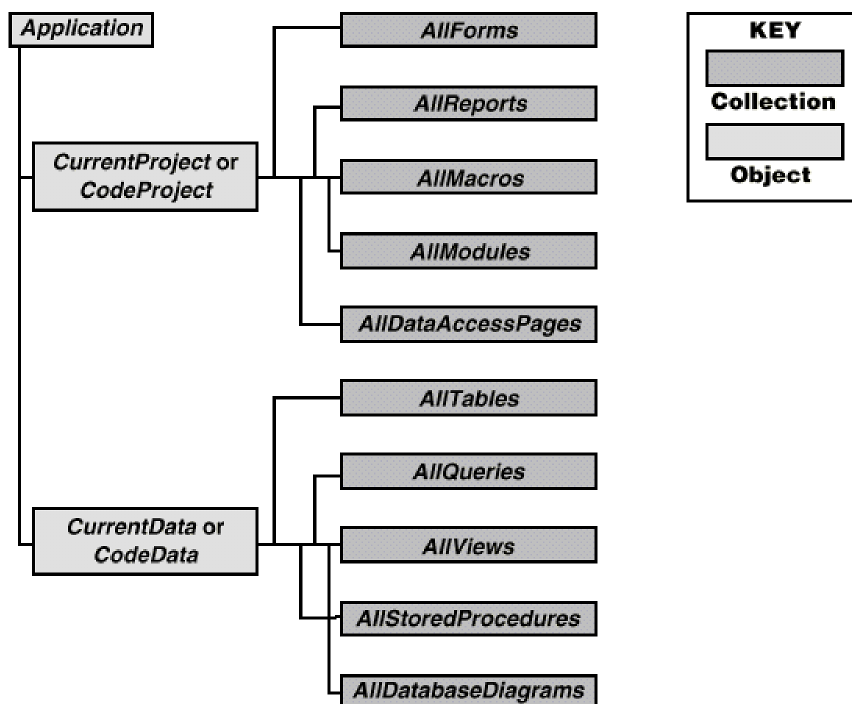
כשאתה פוגש באובייקט מסוג AccessObject שהיישום שלך זקוק לפרטים נוספים אודותיו, באפשרותך לנצל את המאפיינים IsLoaded ו-Name כדי לבדוק את מאפייני האובייקט שאליו מצביע האובייקט AccessObject. אוספי אובייקטים פתוחים אלה כוללים מערך מקיף של מאפיינים ושיטות שאינם זמינים באוספי All.

אובייקטי AccessObject כוללים מאפיין מהסוג Type שמתאר את הסוג של AccessObject ולא את הסוג של אובייקט מסד הנתונים. ערך המאפיין Type של כל חבר באוסף AllModules הוא 5. ערך זה מייחד חבר מסוג AccessObject באוסף AllModules מחבר אחר באוסף AllForms, שערך המאפיין Type שלו הוא 2. בכל מקרה, לא תוכל לקבוע אם אתה מטפל במודול מחלקה או במודול מחלקת טופס על ידי בדיקה פשוטה של החבר AccessObject באוסף All. עליך לבדוק את המאפיין Type של אובייקט Module ואת המאפיין HasModule של אובייקט Form.

האוספים All

קיימים שבעה אוספים שמתחילים ב-All, פרט לאוספים AllModules, AllForms ו-AllReports. קבוצת עשרת האוספים (ראה תרשים 7.5) מתחלקת באופן טבעי לשתי קבוצות בנות חמישה אוספים כל אחת. האוספים AllForms, AllReports, AllMacros

ו-AllDataAccessPages הם חברים באובייקטים CurrentProject ו-CodeProject באובייקט Aplication של Access. האוספים AllTables, AllQueries, AllViews, AllStoredProcedures ו-AllDatabaseDiagrams הם חברים באובייקטים CurrentData ו-CodeData באובייקט Application של Access. כשאתה מציין אובייקטים מסוג AccessObject באוסף כלשהו מתוך עשרת האוספים, עליך להגדיר הפניה שמצביעה על האובייקט הקודם הנכון. אם לא תקפיד לעשות זאת, תקבל שגיאה.



תרשים 7.5: עשרת האוספים All והקרבה ההיררכית ביניהם לבין האובייקטים Project ו-Data.

סוג הקובץ של Access מעניק זמינות מוגבלת לאוספים AllQueries, AllViews, AllStoredProcedures ו-AllDatabaseDiagrams. כזכור, פרויקט של Access יכול להופיע בקובץ המקובל בתבנית mdb.*, או בתבנית החדשה של Access 2000, adp.* (פרק 4 מציג פרטים בסיסיים של קובץ adp.*, ופרק 12 עוסק בכך בהרחבה). האוסף AllQueries זמין בקבצי mdb.*, אך לא בקבצי adp.*. לעומת זאת, באפשרותך לטפל באוספים AllViews, AllStoredProcedures ו-AllDatabaseDiagrams מתוך קבצים מסוג adp.*, אך לא מקבצים מסוג mdb.*. היישומים יכולים עדיין להפנות לתצוגות ושגרות מאוחסנות בקבצים מסוג mdb.* באמצעות ספריית האובייקט ADOX.



הערה:

ייתכן שתהית מדוע Access 2000 מאפשר לטפל באוספים Vies ו-Procedurs מתוך קבצים מסוג *.mdb, אך אינו מאפשר טיפול באוספים AllViews ו-AllStoredProcedures בקבצים מסוג *.mdb. שני האוספים האחרונים לא נחשפו לקבצים מסוג *.mdb עקב הצורך למלא דרישות שקדימותן גבוהה יותר. טיפול באוספים AllViews ו-AllStoredProcedures בקבצים אלה יהיה אפשרי במהדורה עתידית של Access, או בעדכון מהדורתו הנוכחית.

מספור חברי האוסף All

שלוש השגרות שלפניך מפגינות רמת דמיון גבוהה בתכנות אוספים שונים מהסוג All. השיגרה הראשונה מבצעת מספור פשוט של כל המודולים בפרויקט הנוכחי. שים לב כי היא מצהירה תחילה על obj1 בתור סוג AccessObject, מכיון שהיא מקבלת את זהות האלמנטים שבאוסף AllModules, שכולל אובייקטים מהסוג AccessObject. שים לב גם שלולאת המספור עוברת על האוסף AllModules, אך הקוד מגיע לאוסף זה באמצעות האובייקטים Application ו-CurrentProject.

```
Sub EnumerateAllModules()
Dim obj1 As AccessObject
   For Each obj1 In Application.CurrentProject.AllModules
      Debug.Print obj1.Name & vbTab & obj1.Type & vbTab & obj1.IsLoaded
   Next obj1
End Sub

Sub EnumerateAllForms()
Dim obj1 As AccessObject
   For Each obj1 In Application.CurrentProject.AllForms
      Debug.Print obj1.Name & vbTab & obj1.Type & vbTab & obj1.IsLoaded
   Next obj1
End Sub

Sub EnumerateAllTables()
Dim obj1 As AccessObject
   For Each obj1 In Application.CurrentData.AllTables
      Debug.Print obj1.Name & vbTab & obj1.Type & vbTab & obj1.IsLoaded
   Next obj1
End Sub
```

מבנה השגרות EnumerateAllForms ו-EnumerateAllTables דומה לזה של השיגרה EnumerateModules. יחד עם זאת, עליך לשים לב להבדלים משמעותיים אחדים בין תוכן השגרות. ראשית, האוסף הספציפי AccessObject משתנה מ-AllModules ל-AllForms בשיגרה אחת ול-AllTables בשיגרה האחרת. שנית, הנתיב אל האוסף

AllTables עובר דרך האובייקט CurrentData, ולא דרך האובייקט CurrentProject. לו היה עלינו להחליף את האוסף AllTables ל-AllViews או ל-AllStoredProcedures, הקוד היה מסוגל לפעול גם בקובץ מסוג adp.*, אך לא בקובץ מסוג mdb.*.

התאמה לסוגי הקבצים mdb.* ו-adp.*

המאפיין ProjectType של האובייקט CurrentObject מאפשר לגלות את סוג הקובץ בו מטפלים, mdb.* או adp.*. הדבר מאפשר כתיבת שגרות בודדות שמתאימות לסביבה בה הן אמורות לפעול. הדוגמה שלפניך מציגה את שמן של כל התצוגות והשגרות המאוחסנות בקובץ adp.*, אך במקרה הצורך היא עוברת להצגת כל השאילתות שבקובץ mdb.*. כפי שניתן לראות, התחבולה היחידה הדרושה כאן היא לבדוק את ערך המאפיין ProjectType. המאפיין Type של AccessObject מאפשר להבדיל אובייקטים של תצוגות שערך הסוג שלהם 7, מאובייקטים שמצביעים אל שגרות מאוחסנות, שערך הסוג שלהן הוא 9.

```
Sub EnumerateAllViews2()
Dim obj1 As AccessObject, dbs1 As Object
    Set dbs1 = Application.CurrentData
    If Application.CurrentProject.ProjectType = acADP Then
        For Each obj1 In dbs1.AllViews
            Debug.Print obj1.Name & vbTab & obj1.Type & vbTab & obj1.IsLoaded
        Next obj1
        For Each obj1 In dbs1.AllStoredProcedures
            Debug.Print obj1.Name & vbTab & obj1.Type & vbTab & obj1.IsLoaded
        Next obj1
    Else
        For Each obj1 In dbs1.AllQueries
            Debug.Print obj1.Name & vbTab & obj1.Type & vbTab & obj1.IsLoaded
        Next obj1
    End If
End Sub
```

AllForms ו-AllModules

הדוגמה הבאה מנצלת את האוספים All והאוספים המתאימים של מודולים וטפסים פתוחים כדי לפתח רשימה של כל המודולים (לפי סוג) ומודולי המחלקה של טפסים בפרויקט. הואיל ומאפיין הציון של מודולי מחלקה סטנדרטיים שונה מזה של מודולי מחלקה של טפסים, הקוד מחייב בדיקה של ביטויים שונים של מודולי מחלקה סטנדרטיים לעומת מודולי מחלקה של טפסים.

כזכור, למודולים יש מאפיין Type, אך לטפסים יש מאפיין HasModule. על הקוד לעבור במחזוריות על חברי האוספים AllModules ו-AllForms, מכיון שחלק מהמודולים והטפסים, או אפילו כולם, עלולים להיות סגורים. בודקים את המצב IsLoaded של האובייקטים AccessObject ב-AllModules וב-AllForms כדי לקבוע אם יש צורך לפתוח מודול או קובץ טרם הערכת סוג המודול שלו, או כדי לקבוע אם לטופס יש מודול מחלקה. השיגרה סוגרת מחדש טפסים ומודולים לאחר שבדקה אותם.

```
Sub ListAllModulesByTypeAndClassForms()
Dim obj1 As AccessObject, dbs1 As Object
Dim mod1 As Module, frm1 As Form

    Set dbs1 = Application.CurrentProject

' Search for open AccessObject objects in AllModules collection.
' Open and reclose those that are not open.

    For Each obj1 In dbs1.AllModules
        If obj1.IsLoaded = True Then
            ListTypeOfModule obj1.Name
        Else
            DoCmd.OpenModule obj1.Name
            ListTypeOfModule obj1.Name
            DoCmd.Close acModule, obj1.Name
        End If
    Next obj1

' Search for open AccessObject objects in AllForms collection.
' Open and reclose those that are not open.

    For Each obj1 In dbs1.AllForms
        If obj1.IsLoaded Then
            DoesFormHaveModule obj1.Name
        Else
            DoCmd.openform obj1.Name
            DoesFormHaveModule obj1.Name
            DoCmd.Close acForm, obj1.Name
        End If
    Next obj1
End Sub
```

```
Sub ListTypeOfModule(modname)
Dim strType As String
' Decode module Type value.
    If Modules(modname).Type = 0 Then
        strType = "Standard Module"
    Else
        strType = "Class Module"
    End If
 ' Print module name and type.
    Debug.Print Modules(modname).Name & vbTab & strType
End Sub

Sub DoesFormHaveModule(frmname)
' Only print form name if it has a module.
    If Forms(frmname).HasModule = True Then
        Debug.Print frmname & vbTab & "Form Class Module"
    End If
End Sub
```

עריכת מודולים באמצעות קוד

מאחר שניתן לבצע משימות רבות מאוד באמצעות מודולים סטנדרטיים, מודולי מחלקה עצמאיים ומודולי מחלקה של טפסים, סביר שהיישומים שתיצור יכילו מודולים רבים מסוגים אלה. בבוא הזמן יהיה צורך לתחזק אותם. מטלת תחזוקה שכיחה היא הדרישה להוסיף או למחוק שורת קוד אחת או יותר בקבוצת מודולים. סעיף זה מתאר כיצד להוסיף ולמחוק שורה בכל מודולי המחלקה הסטנדרטיים והעצמאיים ולאחר מכן מסביר כיצד עושים זאת במודולי מחלקת טופס. הקוד של מודולי מחלקה סטנדרטיים ועצמאיים מאוחסן בצורה שונה מזו של הקוד של מודולי מחלקת טופס, ולכן יש הבדלים מסוימים בין התהליכים.

גישות עריכה

האובייקט Module כולל מערך שיטות ומאפיינים שיכולים לסייע בעריכת מודולים באמצעות קוד. הדוגמאות בסעיף זה מבוססות על השיטות InsertLines, Find ו-DeleteLines. שיטות אלו מטפלות במודולים סטנדרטיים ובמודולי מחלקה, כולל מודולי מחלקה עצמאיים וגם מודולי מחלקת דוח וטופס. שיטות אלו מהוות קבוצת משנה של שיטות ומאפיינים שתומכות בניהול תוכן מודול באמצעות קוד.

השיטה InsertLines יחד עם האובייקט Module משמשים להוספת שורה אחת או יותר במודול. מספר שורות המודול מתחיל ב-1 ומגיע עד לערך שמכיל המאפיין CountOfLines של המודול. השיטה קולטת את מספר השורה וארגומנט מחרוזת. אם

עליך להוסיף שורות רבות למודול, הוסף את הקבועים vbCrLf לביטוי המחרוזת שמייצג את ארגומנט המחרוזת של השיטה. כשמוסיפים שורות באמצעות שיטה זו, היא עוברת לשורות הבאות במודול.

השיטה Find מחפשת מחרוזת טקסט במודול. היא מחזירה ערך True כשהיא מוצאת את הטקסט המבוקש, או False במקרה שלא. אם ידוע לך בדיוק היכן נמצא קטע טקסט מסוים, באפשרותך להגדיר עמודה ושורת פתיחה, ועמודה ושורת סיום. אם אינך יודע את מיקומו של קטע הטקסט במודול, השאר את הארגומנטים של מיקום הטקסט ריקים, והפונקציה תחזיר את ערכי הטקסט שאתה מחפש במודול. באפשרותך גם להקצות חיפוש לפי תבנית וחיפוש תלוי רישיות.

השיטה DeleteLines מסירה שורת טקסט אחת או יותר ממודול. השיטה קולטת שני ארגומנטים: שורת פתיחה ומספר כולל של שורות המיועדות להסרה מהמודול. באפשרותך לנצל את השיטה Find כדי לחפש טקסט במודול. אם תעשה זאת, תוכל לבסס את הבקשה של השיטה DeleteLines על ערך ההחזרה של השיטה Find.

הוספת טקסט למודולים

השגרות שלפניך משלבות את האוספים AllModules ו-Modules כדי לערוך את הטקסט באוסף של מודולים. ליתר דיוק, הן מוסיפות שורת הערה בתחילת כל מודול כדי להכריז עליו שהוא מודול סטנדרטי או מודול מחלקה. השיגרה EnumerateAllModulestoInsert עוברת בלולאה על חברי האוסף AllModules וקוראת לשיגרה האחרת, שמעדכנת בפועל את מודולי היעד. השיגרה InsertIntoModules דורשת מודול פתוח, ולכן השיגרה הראשונה פותחת את המודול אם אינו פתוח עדיין. לאחר שהשיגרה השנייה מחזירה את השליטה לשיגרה הראשונה, היא סוגרת את המודול כדי להחזירו למצבו המקורי.

```
Sub EnumerateAllModulestoInsert()
Dim obj1 As AccessObject
' Loop through AllModules members.
' If module is open, call sub to insert lines;
' else open module first, then close afterwards.
   For Each obj1 In Application.CurrentProject.AllModules
      If obj1.IsLoaded = True Then
         InsertIntoModules obj1.Name
      Else
         DoCmd.OpenModule obj1.Name
         InsertIntoModules obj1.Name
         DoCmd.Close acModule, obj1.Name, acSaveYes
      End If
   Next obj1
End Sub
```

```
Sub InsertIntoModules(modname)
Dim strType As String, mod1 As Module
    Set mod1 = Modules(modname)

' Detect module type to determine which string to insert.
    If mod1.Type = 0 Then
        strType = "'Standard Module"
    Else
        strType = "'Class Module"
    End If
    mod1.InsertLines 1, strType
    Set mod1 = Nothing

End Sub
```

השיגרה InsertIntoModules קולטת ארגומנט יחיד – שם המודול אותו יש לערוך. השיגרה אינה מבצעת מעבר מחזורי כלשהו, מכיון שהשיגרה הראשונה קוראת לה פעם אחת לכל חבר באוסף AllModules. השיגרה פותחת בהגדרת הפניה אל המודול ששמו הועבר בצורת ארגומנט. לאחר מכן היא קובעת את סוג המודול שאליו ההפניה מפנה ומכניסה ערך משתנה מחרוזת להערה שמציינת את שם סוג המודול. לאחר שנקבע הטקסט שיש להוסיף, השיגרה מפעילה את השיטה InsertLines על המודול שאליו הוגדרה ההפניה.

מחיקת טקסט ממודולים

שתי השגרות שלפניך מוחקות שורה מתוך שיגרה. לאמיתו של דבר, הן מסירות את השורה שהוספה באמצעות זוג השגרות הקודמות. עיצוב שתי השגרות הבאות גמיש דיו, ולכן באפשרותך להרחיבן בקלות כך שיכילו מחיקה של שורות נבחרות רבות מתוך קבוצת מודולים כלשהי.

השגרות מבוססות על לוגיקה כללית זהה לזו של זוג השגרות הקודם בהבדל עיקרי אחד: זוג זה מנצל את השיטות Find ו-Delete כדי להסיר טקסט במקום השיטה InsertLines. לשיטה Find חשיבות קריטית כשמתכוננים להפעיל את השיטה DeleteLines, מכיון שהיא מאפשרת לקוד לקבוע את מציאותו של טקסט כלשהו במודול בטרם תמחק אותו. במקרה שלפנינו, השיטה Find מחפשת את המילה Module ברצף 40 התווים הראשונים בשורה הראשונה. השיגרה DeletefromModules מפעילה את השיטה DeleteLines כדי למחוק שורה אחת החל מהשורה הראשונה של המודול. השיטה DeleteLines מסירה שורות בצורה בלתי מותנית. יחד עם זאת, ניתן להפעיל בצורה ידנית את הפונקציה Undo Delete כדי לשחזר טקסט שהוסר.

```
Sub EnumerateAllModulestoDelete()
Dim obj1 As AccessObject, dbs As Object
Dim mod1 As Module, frm1 As Form

' Loop through AllModules members.
' If module is open, call sub to delete line;
' else open module first, then close afterwards.
    For Each obj1 In Application.CurrentProject.AllModules
        If obj1.IsLoaded = True Then
            DeletefromModules obj1.Name
        Else
            DoCmd.OpenModule obj1.Name
            DeletefromModules obj1.Name
            DoCmd.Close acModule, obj1.Name
        End If
    Next obj1
End Sub

Sub DeletefromModules(modname)
Dim mod1 As Module
    Set mod1 = Modules(modname)

' Delete first line if first 40 characters contain "Module".
    If mod1.Find("Module", 1, 1, 1, 40) = True Then
        mod1.DeleteLines 1, 1
    End If
    Set mod1 = Nothing
End Sub
```

הוספת טקסט למודולי מחלקת טופס

שתי השגרות שלפניך מוסיפות שורה בתחילת כל מודול מחלקת טופס, ובה הערה כי מדובר במודול מחלקה. במקום לעבור בלולאה על כל האוסף AllModules, השיגרה הראשונה עוברת על האוסף AllForms. לכל חבר באוסף, היא קוראת לשיגרה InsertIntoForms.

שיגרה שנייה זו מעריכה אם שם הטופס שהועבר הוא מודול מחלקה. אם הוא אכן כזה, השיגרה מגדירה הפניה למודול שברקע הטופס. על ידי כך היא חושפת את המודול. השיגרה מסתיימת בהוספת שורת ההערה למודול והגדרת הפניה אל Nothing כדי לשחרר את המשאבים.

```
Sub EnumerateAllFormsToInsert()
Dim obj1 As AccessObject

' Loop through AllForms members;
' if form is loaded invoke module to insert line,
' else open form first and then close afterwards.
   For Each obj1 In Application.CurrentProject.AllForms
      If obj1.IsLoaded Then
         InsertIntoForms obj1.Name
      Else
         DoCmd.openform obj1.Name
         InsertIntoForms obj1.Name
         DoCmd.Close acForm, obj1.Name, acSaveYes
      End If
   Next obj1
End Sub

Sub InsertIntoForms(frmname)
Dim mod1 As Module, strType As String

' If Form has module, set reference to it and insert line into the module.
' Free reference resource when done.
   If Forms(frmname).HasModule = True Then
      Set mod1 = Forms(frmname).Module
      strType = "'Form Class Module"
      mod1.InsertLines 1, strType
      Set mod1 = Nothing
   End If
End Sub
```

מחיקת טקסט ממודולי מחלקת טופס

שתי השגרות שלפניך מסירות את שורת ההערה Class Module מהשורה הראשונה של המודולים שברקע הטפסים. כפי שניתן להבחין, זוג השגרות מאמץ את התנהגותם של מרכיבים קריטיים מזוג השגרות הקודם בנוגע להוספה ומחיקה של שורות. זוג השגרות עובר בצורה מחזורית על האוסף AllForms, כמו זוג השגרות שהוסיף את שורת ההערה בתחילת כל מודולי מחלקת הטופס של הפרויקט.

יחד עם זאת, השיגרה השנייה בזוג זה מנצלת את השיטות Find ו-DeleteLines כדי להסיר את השורה הראשונה במודול, אם המילה Module מופיעה בתחום 40 התווים הראשונים בשורה. בכך היא דומה לשיגרה שמוחקת שורות מהאוסף Modules.

```
Sub EnumerateAllFormstoDelete()
Dim obj1 As AccessObject

' Loop through AllForms members;
' if form is loaded invoke module to remove line,
' else open form first and then close afterwards.
   For Each obj1 In Application.CurrentProject.AllForms
      If obj1.IsLoaded Then
         DeletefromForms obj1.Name
      Else
         DoCmd.openform obj1.Name
         DeletefromForms obj1.Name
         DoCmd.Close acForm, obj1.Name, acSaveYes
      End If
   Next obj1
End Sub

Sub DeletefromForms(frmname)
Dim mod1 As Module, strType As String

' If form has module, then check contents of first line
' for "Module", and delete the first line if it is present.
' Free module reference resource when done.
   If Forms(frmname).HasModule = True Then
      Set mod1 = Forms(frmname).Module
      If mod1.Find("Module", 1, 1, 1, 40) = True Then
         mod1.DeleteLines 1, 1
      End If
      Set mod1 = Nothing
   End If
End Sub
```

8

אובייקטי Microsoft Office

כמרכיב של Microsoft Office 2000, חוֻלק Microsoft Access 2000 מבחר אובייקטים עם שאר יישומי Office. אובייקטים אלה מאפשרים ביצוע משימות כמו חיפוש קבצים, טיפול בתכונת העזרה Office Assistant, עריכת שינויים בתפריטים ובסרגלי הכלים הסטנדרטיים, ופיתוח של תפריטים וסרגלי כלים מותאמים אישית. נוסף לכך, הבקיאות שתפתח בתכנות האובייקטים הללו ב-Access תשמש אותך גם ביישומי Office האחרים – Microsoft Excel, Microsoft Word, ואפילו Microsoft FrontPage. רוב האובייקטים פועלים בכל יישומי Office.

פרק זה פותח בסקירה כללית על האובייקטים של Office, ולאחר מכן מתמקד בשלושה אובייקטים: FileSearch, המשמש לניהול תוכניתי של חיפושי קבצים; Assistant, המספק ממשק תוכניתי לתכונה Office Assistant במסגרת Office; וכן CommandBars, המשמש ליצירת תפריטים וסרגלי כלים מותאמים אישית.

שימוש באובייקטים המשותפים של Office

האובייקטים המשותפים של Office, המופיעים בטבלה שבעמוד הבא, תומכים במספר תחומי פונקציונליות חשובים. חלק ממבני האובייקטים הללו אינם זמינים בכל רכיבי Office. הטבלה מתארת את המבנים, מציינת אם זמינותם מוגבלת לרכיבי Office מסוימים, ומפנה למידע נוסף אודותיהם בספר זה. לקבלת עזרה מקוונת וכדי לעבוד עם אובייקטים אלה, פנה ל-Microsoft Office 9 Object Library. תוכל לעשות זאת באמצעות תפריט **Tools** (כלים) בעורך Visual Basic (VBE), או באופן תוכניתי (כפי שיודגם בהמשך פרק זה).

אובייקטים משותפים ב-Office

האובייקט	תיאור	הערות
CommandBar	השתמש באובייקט זה ובאוספים של יצירת סרגלי כלים, שורת התפריטים ותפריטי קיצור, או להכנסת שינויים בפריטים אלה. באפשרותך להוסיף שינויים בזמן-עיצוב באופן ידני או באמצעות קוד VBA. שינויים בזמן-ריצה אפשריים בעזרת קוד VBA בלבד.	ראה דיון נוסף ודוגמאות בהמשך פרק זה.
Assistant	השתמש באובייקט זה לתמיכה בדרישות עזרה מותאמות אישית עבור Office Assistant והבלון של Office Assistant.	ראה דיון נוסף ודוגמאות בהמשך פרק זה.
FileSearch	השתמש באובייקט זה לייצג הפונקציונליות של תיבת הדו-שיח **פתיחה** (Open).	ראה דיון נוסף ודוגמאות בהמשך פרק זה.
COMAddin	זהו ייצוג של תוספת COM ב-Access ובישומי מארח אחרים ב-Office.	עיין במדריך Microsoft Office 2000 Programmer's Guide.
LanguageSettings	אובייקט זה מאפשר החזרה תכונתיות של מידע אודות התקנת שפות, ממשק המשתמש וההגדרות העזרה.	מנהל מעקב אחר נתוני זיהוי מקומיים בפריסה בינלאומית של Office.
AnswerWizard	השתמש באובייקט זה לטיפול תכונתיי באשף התשובות.	כולל מאפיינים ושיטות לטיפול בקבצים שמחזירי אשף התשובות.
DocumentProperty	אובייקט זה מייצג מאפיין מוכלל או מותאם אישית של מסמך Office. לבחירתך עד 28 מאפיינים מוכללים, הכוללים תכונות מסמך כמו כותרת, מחבר, הערות, תאריך הדפסה אחרונה, שעת שמירה אחרונה וזמן עריכה כולל. אובייקט זה תומך גם בתכונות מסמך מותאמות אישית.	מיועד למסמכי Word, לחוברות עבודה ב-Excel ולמצגות ב-Microsoft PowerPoint בלבד.
HTMLProject	אובייקט זה הוא פרויקט ברמה העליונה בעורך ה-Script של Microsoft. השתמש באוסף HTMLProjctItem למעקב אחר מסמכי HTML במסגרת פרויקט.	מיועד למסמכי Word, לחוברות עבודה ב-Excel ולמצגות ב-Microsoft PowerPoint בלבד.

האובייקט	תיאור	הערות
Script	אובייקט זה מייצג בלוק של Script בעורך ה-Script.	מיועד למסמכי Word, לחוברות עבודה ב-Excel ולמצגות ב-Microsoft PowerPoint בלבד.
WetPageFont	אובייקט זה מייצג את גופן ברירת המחדל בעת שמירת מסמך כדף Web.	מיועד למסמכי Word, לחוברות עבודה ב-Excel ולמצגות ב-Microsoft PowerPoint בלבד.

מאפייני Access Database

שלא כמו ב-Word, Excel ו-PowerPoint, חסר ב-Access האובייקט המשותף DocumentProperty, אולם הוא מעמיד לרשותך מידע דומה בעזרת שלושה אובייקטים של Document: MSysDB, SummaryInfo ו-UserDefined. אובייקטים אלה זמינים אך ורק באמצעות Database Container של Data Access Object (DAO). האובייקט SummaryInfo מכיל את כל המאפיינים המופיעים בכרטיסיה **תקציר** (Summary) שבתיבת הדו-שיח **מאפייני מסד נתונים** (Database Properties). האובייקט UserDefined מכיל את כל המאפיינים הכלולים בכרטיסיה **התאמה אישית** (Custom) באותה תיבת דו-שיח. האובייקט MSysDB מכיל את כל המאפיינים המוגדרים באמצעות האפשרות **הפעלה** (Startup) שבתפריט **כלים** (Tools), במסד נתונים.

הדוגמה הבאה מונה את אוסף המאפיינים של כל אובייקט DAO Database Container:

```
Sub enumDBProps()
Dim db As Database, p As DAO.Property

' Set reference to current database.
   Set db = CurrentDb

' Print heading for results.
   Debug.Print "User defined properties"
   Debug.Print "======================="
' Iterate through UserDefined database properties.
   For Each p In db.Containers!Databases.Documents!UserDefined.Properties
      Debug.Print p.Name, p.Value
   Next

' Print heading for results.
   Debug.Print
   Debug.Print "Summary Properties"
   Debug.Print "=================="
```

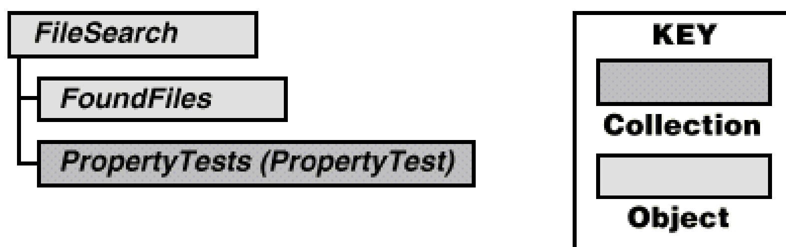
```
' Iterate through SummaryInfo database properties.
   For Each p In db.Containers!Databases.Documents!SummaryInfo.Properties
      Debug.Print p.Name, p.Value
   Next

' Print heading for results.
   Debug.Print
   Debug.Print "MSysDB Properties"
   Debug.Print "================="

' Iterate through MSysDB database properties.
   For Each p In db.Containers!Databases. Documents!MSysDB.Properties
      Debug.Print p.Name, p.Value
   Next
End Sub
```

האובייקט FileSearch

השתמש בתבנית האובייקט FileSearch (המוצגת בתרשים 8.1) כדי לשלב חיפושי קבצים ביישומיך. תוכל לחפש בכוננים הקשיחים של המחשב או בכוננים של מחשבים המקושרים ברשת תקשורת מקומית (LAN). אובייקט זה חושף את הפונקציונליות של תיבות הדו-שיח **פתיחה** (Open) **וחיפוש** (Find). כפי שמתואר בתרשים 8.1, האובייקט FoundFiles והאוסף PropertyTests תלויים מבחינה היררכית באובייקט FileSearch.



תרשים 8.1: האובייקט FileSearch הוא אחד האובייקטים המשותפים של Office.

באפשרותך להגדיר חיפוש קובץ בשתי דרכים בסיסיות, וכל גישה מתאימה לאפשרויות שונות בתיבת הדו-שיח **פתיחה**.

- חיפוש לפי קריטריון יחיד (שם קובץ או תבנית, סוג קובץ, או נתיב).
- חיפוש תוכניתי לפי קריטריונים מרובים, בעזרת האוסף PropertyTests.

בגישה השנייה, תוכל להגדיר ארגומנטים אשר ישמשו קלט לתיבת הדו-שיח **פתיחה**. השתמש בתיבת דו-שיח זאת כדי לציין, תנאים מרובים לחיפוש, וכן כללים לשִרשור, כגון האופרטורים And או Or. השתמש באובייקט FoundFiles כדי למנות את הקבוצה המוחזרת בשתי הגישות.

לאובייקט FileSearch שתי שיטות: NewSearch ו-Execute. השיטה NewSearch מאפסת את כל מאפייני FileSearch ומחזירה אותם לערכי ברירת המחדל. בנקודה זאת תוכל לערוך את המאפיינים שנדרשים בהם ערכים מיוחדים עבור חיפוש קובץ מסוים. אם לא תפעיל את השיטה NewSearch בתחילת המפרט לחיפוש, החיפוש החדש יירש את הגדרות המאפיינים של החיפוש הקודם.

באפשרותך להפעיל את השיטה Execute כדי להפעיל בחיפוש קובץ לאחר שהגדרת אותו. שיטה זאת מקבלת מספר ארגומנטים השולטים בסידור שמות הקובץ באובייקט FoundFiles, ובאפשרות לעדכן, או שלא לעדכן, את אינדקס הקבצים לפני עריכת חיפוש חדש. ערך ההחזרה בשיטה זאת הוא מספר שמות הקובץ התואמים את מפרט החיפוש.

ניהול חיפוש קובץ בסיסי

רבים ממאפייני האובייקט FileSearch מאפשרים מפרט חיפוש גמיש. הקוד שבדוגמה הפשוטה הבאה מגדיר חיפוש ומאחזר את הקבוצה המוחזרת שלו. הוא יוצר מופע של האובייקט באמצעות המאפיין FileSearch של האובייקט Application. לאחר מכן הוא מחזיר את כל הגדרות המאפיין FileSearch לערכי ברירת המחדל על ידי הפעלת השיטה NewSearch. בשלב הבא הקוד מקצה את המאפיינים LookIn ו-FileName, המגדירים היכן לחפש ומה לחפש. מנגנון הבדיקה לחיפוש זה מכיל סדרה של קבצי mdb. עם שמות כמו Chapter1 ו-Chapter2. המאפיין SearchSubFolders מקבל ערך בוליאני המציין אם יש להגביל את החיפוש לתיקיה הנוכחית, או להרחיב אותו לתיקיות המשנה שבהגדרת המאפיין LookIn.

```
Sub FileSearch1()
' Search in My Documents folder and its subfolders for Chapter*.mdb.
With Application.FileSearch
' Start a new search.
   .NewSearch
' Set search criteria.
   .LookIn = "C:\My Documents"
   .FileName = "Chapter*.mdb"
   .SearchSubFolders = True
End With
```

```
With Application.FileSearch
' Execute the Search.
    If .Execute() > 0 Then
        MsgBox "There were " & .FoundFiles.Count & " file(s) found."
' Display names of all found files.
        For i = 1 To .FoundFiles.Count
            MsgBox .FoundFiles(i)
        Next i
    Else
' If no files found, say so.
        MsgBox "There were no files found."
    End If
End With
End Sub
```

לאחר שהשיגרה יוצרת מפרט לחיפוש, היא מפעילה את השיטה Execute עבור האובייקט FileSearch. לשיטה זאת ערך מוחזר המציין את מספר הקבצים העונים על התנאים שהוגדרו לחיפוש. אם הערך הוא 0, החיפוש לא הניב שמות קובץ תואמים והשיגרה תציג הודעה המציינת כי לא נמצאו קבצים. אם נמצא קובץ תואם אחד, או יותר, לפי התנאים, השיגרה תציג את המאפיין Count של האובייקט FoundFiles בטרם תציג את השמות ב-FoundFiles, אחד לאחד.

מיון הקבוצה המוחזרת

דוגמת הקוד הבאה ממיינת את הקבוצה המוחזרת של חיפוש לפי גודל קובץ. שני הפרמטרים הראשונים לשיטה Execute קובעים את תנאי המיון וסדר המיון, בהתאמה. השמות הקבועים בפרמטר הראשון מציינים את המשתנה שלפיו יש למיין את שמות הקבצים המוחזרים. קבועים אלה הם msoSortByFileName, msoSortByFileType, msoSortByLastModified ו-msoSortBySize. הפרמטר השני בשיטה Execute מציין סדר עולה או יורד. דוגמת הקוד מגדירה חיפוש שהמיון בו ייעשה לפי גודל הקובץ, בסדר יורד, להבדיל מהדוגמה הקודמת, שבה הוחזרו התוצאות בסדר עולה, שהוא ברירת המחדל, לפי שמות הקבצים.

```
Sub FileSearch2()
Dim sngMB As Single
' Search in My Documents folder and its subfolders for Chapter*.mdb.
With Application.FileSearch
' Start a new search.
    .NewSearch
' Set search criteria.
    .LookIn = "C:\My Documents"
    .FileName = "Chapter*.mdb"
    .SearchSubFolders = True
End With
```

```
With Application.FileSearch

' Return found files in descending order by file size.
    If .Execute(msoSortBySize, msoSortOrderDescending) > 0 Then
        MsgBox "There were " & .FoundFiles.Count & " file(s) found."
        For i = 1 To .FoundFiles.Count

' Compute file size in MB and display with filename.
            sngMB = FileLen(.FoundFiles(i)) / (1024 ^ 2)
            MsgBox .FoundFiles(i) & vbCrLf & vbTab & _
                "Filesize (MB): " & Round(CDec(sngMB), 3)
        Next i

    Else
' If no files found, say so.
        MsgBox "There were no files found."
    End If

End With

End Sub
```

תיבת ההודעה המציגה את הקבוצה המוחזרת מראה גם את גודל הקובץ ושמו. לקביעת גודלו של קובץ יש להעביר את האובייקט FoundFiles אל הפונקציה FileLen. גודלי הקבצים מעוגלים אל אלפית המ״ב הקרובה ביותר.



הערה:

הפונקציה Round היא פונקציה חדשה בגירסה VBA 6 של Visual Basic for Applications. כדי לקבל תוצאות עקביות עם פונקציה זאת, עליך להעביר תחילה את הארגומנט שלה אל הפונקציה CDec. בדוגמה הקודמת נעשה שימוש בתחביר כזה (פנה לפרק 1 לקבלת מידע על הפונקציה CDec).

חיפוש המבוסס על תוכן הקובץ

גם כאשר מדובר בחיפוש פשוט, כמו בשתי הדוגמאות הקודמות, תוכל לחפש קטע טקסט מוגדר במסמך, או באובייקט DocumentProperty שלו. בדוגמה הבאה מתבצעת פעולה זאת. כאן משתמשים במאפיין TextOrProperty של האובייקט FileSearch כדי לסמן מחרוזת טקסט מבוקשת בגוף הקובץ או באוסף Properties של הקובץ. שים לב, שניתן לציין תיקיות במחשבים מרוחקים בעזרת המוסכמות למתן שמות אחידים (UNC) (הנתיב \\cab233\c\cab מצביע אל התיקיה cab בשם השיתוף c של מחשב ששמו cab233).

```
Sub FileSearch3()
Dim sngStart As Double, sngEnd As Double
Dim i As Integer

' Search in cab folder on linked computer for files containing CAB.
With Application.FileSearch
' Start a new search.
    .NewSearch
' Set search criteria.
    .LookIn = "\\cab233\d\cab\"
    .SearchSubFolders = False
' When searching for text consider restricting the files you search.  *.* takes 300
' seconds, but msoFileTypeWordDocuments takes 22 seconds.
'    .FileName = "*.*"
    .FileType = msoFileTypeWordDocuments
    .TextOrProperty = "CAB"
End With

With Application.FileSearch
' Execute the search.
    sngStart = Now
    If .Execute() > 0 Then
        sngEnd = Now
        Debug.Print DateDiff("s", sngStart, sngEnd)
        MsgBox "There were " & .FoundFiles.Count & " file(s) found."
' Display names of all found files.
        For i = 1 To .FoundFiles.Count
            MsgBox .FoundFiles(i)
        Next I
    Else
' If no files found, say so.
        MsgBox "There were no files found."
    End If
End With

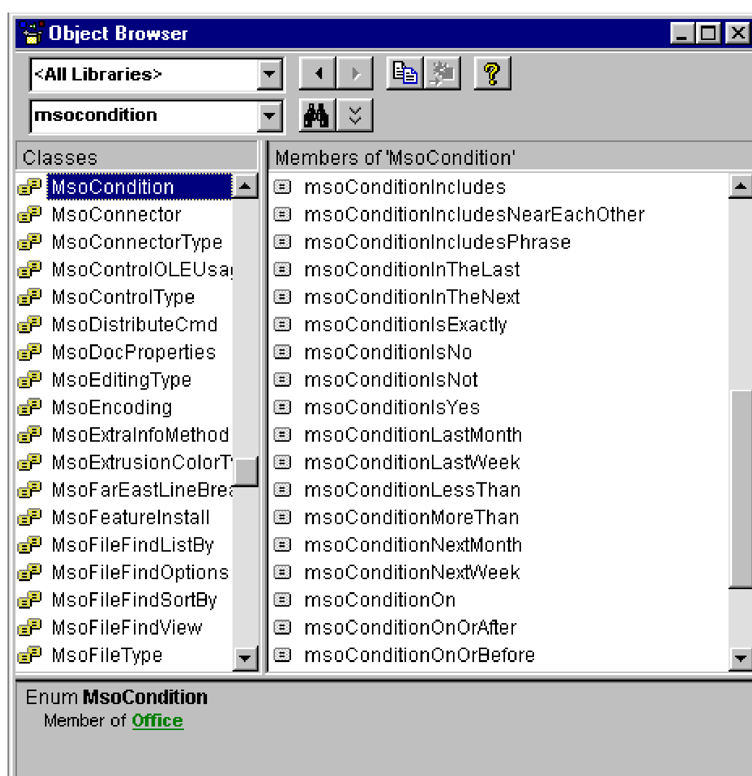
End Sub
```

חיפוש קבצים עלול להימשך זמן רב. תוכל לשפר באורח דרמטי את מהירות הפעולה של השיטה Execute על ידי ציון מאפיין FileSearch מגביל. לדוגמה, דוגמת הקוד הקודמת מאתרת את כל מסמכי Word בתיקיה נתונה, אשר מכילים מחרוזת טקסט מסוימת. השימוש בקבוע msoFileTypeWordDocuments במאפיין FileType מגביל את החיפוש שבדוגמה לקבצים המכילים מסמכי Word בלבד. אל תתפתה לציין *.* עבור המאפיין FileName, מתוך כוונה לסנן את התוצאות המוחזרות, משום שדבר זה יפגע

מאוד במהירות הביצוע. בדוגמה הקודמת, המתייחסת לקבצי התיקיה cab שבמחשב cab233, ההבדל במהירות הפעולה הוא 22 שניות כאשר מופעל הקבוע msoFileType, לעומת 300 שניות להגדרה *.*FileName (שים לב, שדרושות שלוש שורות בלבד למדידת זמן הפעולה – שורה אחת לפני הפעולה Execute, ושתי שורות נוספות מייד לאחריה).

הכללת קריטריוני חיפוש מרובים

תבנית החיפוש המתקדם מאפשרת הכללה של קריטריוני חיפוש מרובים לקבוצה המוחזרת באובייקט FoundFiles. עליך להשתמש בשיטה Add פעמיים, או יותר, כדי לציין קריטריונים מרובים עבור האוסף PropertyTests. על המפרט של כל קריטריון לכלול הגדרות ל-Name ול-Condition.



תרשים 8.2: השתמש בחברים של קבוצת הספירה msoCondition כדי להגדיר תנאים לקריטריונים מתקדמים באוסף PropertyTests של האובייקט FileSearch

השיטה Add יכולה לכלול גם הגדרת Connector, וכן הגדרת Value אחת, או שתיים. כדי לסקור את האפשרויות הזמינות, עליך להציג את איברי המחלקה msoCondition (איור 8.2 מראה קטע כזה). ההגדרות ב-Connector עשויות לקבל ערך אחד, או שניים, שיציינו כיצד לשלב תנאי נתון עם תנאים נוספים. הגדרה זאת מפעילה את

האופרטורים And או Or המשמשים למיזוג של קריטריון מסוים עם קריטריוני חיפוש נוספים. השתמש באופרטור Or לטיפול בקריטריוני החיפוש באופן נפרד, ובאופרטור And כדי לשלב את הקריטריון האמור עם קריטריונים אחרים. And הוא ברירת המחדל להגדרת אופרטור. יחדיו, ההגדרות Condition, Value ו-Connector מספקות פונקציונליות דומה לזו של תיבת הדו-שיח **חיפוש** (Find).

תוכל למנות את חברי האוסף PropertyTests בעזרת לולאת For...Each. כל חבר מהווה קריטריון חיפוש ייחודי. המאפיין Name מזהה את הקריטריון בשעת מניית החברים.

בדוגמה הבאה, שהיא האחרונה לאובייקט FileSearch, שלושה מקטעים. במקטע הראשון מצוינים הקריטריונים, לאחר שהוגדרה הפניה לאובייקט FileSearch. היעד בדוגמה הוא כל קבצי מסד הנתונים בין שני תאריכים. הדוגמה מציגה את התחביר הנכון להפעלת השיטה Add עבור האוסף PropertyTests. הקריטריון הראשון קובע את סוג מסד הנתונים. הקריטריון השני מציין קבצים שהשינוי האחרון בהם נערך בין 1 בינואר 1996 ל- 30 ביוני 1999. ההגדרה msoConnectorOr מציינת שעל הקבצים לקיים את שני הקריטריונים בנפרד כדי להיכלל בקבוצה המוחזרת. אין צורך לציין מאפיין Connector בקריטריון השני, מאחר שהקריטריון השני יאמץ את ערך ברירת המחדל msoConnectorAnd. לפני הצגת הקבוצה המוחזרת, השיגרה מונה את חברי PropertyTests במקטע השני שלה. המקטע השלישי מציג את הקבוצה המוחזרת.

```
Sub Search4()
Dim fs As FileSearch, mystring As String
Dim i As Integer
   Set fs = Application.FileSearch

' Set lookin and subfolder properties.
   With fs
      .NewSearch
      .LookIn = "c:\My Documents"
      .SearchSubFolders = False
   End With

   ' Set a pair of property conditions.
   With fs.PropertyTests
      .Add name:="Files of Type", _
         Condition:=msoConditionFileTypeDatabases, _
         Connector:=msoConnectorOr
      .Add name:="Last Modified", _
         Condition:=msoConditionAnytimeBetween, _
         Value:="1/1/1996", SecondValue:="6/30/1999"
   End With
```

```
' Display property tests.
    For i = 1 To fs.PropertyTests.Count
        With Application.FileSearch.PropertyTests(i)
        mystring = "This is the search criteria: " _
            & " The name is: " & .name & ". The condition is: " & .Condition
        If .Value <> "" Then
            mystring = mystring & ". The value is: " & .Value
            If .SecondValue <> "" Then
                mystring = mystring _
                    & ". The second value is: " _
                    & .SecondValue & ", and the connector is" _
                    & .Connector
            End If
        End If
        MsgBox mystring
        End With
    Next i

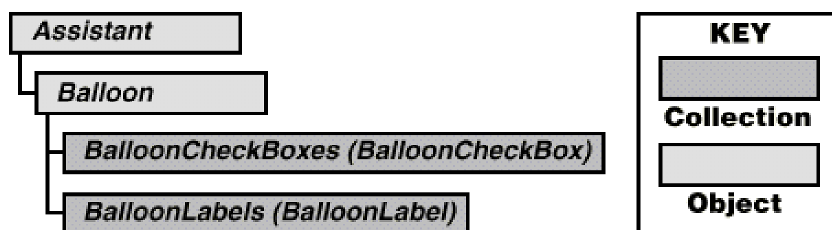
' Display return set from property tests.
    With fs
' Execute the search.
        If .Execute() > 0 Then
            MsgBox "There were " & .FoundFiles.Count & " file(s) found."
' Display names of all found files.
            For i = 1 To .FoundFiles.Count
                MsgBox .FoundFiles(i)
            Next i
        Else
' If no files found, say so.
            MsgBox "There were no files found."
        End If
    End With

End Sub
```

האובייקט Assistant

המסייע של Office הוא תוכנית עזרה ידידותית במסגרת Office. אף שמפתחים ומשתמשים 'כבדים' אולי אינם אוהבים את המסייע, בעיני המשתמש הטיפוסי יש בו יתרונות רבים. קל למדי לתכנת את המסייע, לכן תוכל בנקל להעניק ליישומי Office המותאמים אישית שלך חזות ותחושה דומה לזו של יישומי Office הסטנדרטיים.

מבנה האובייקט Assistant מוצג בתרשים 8.3. ברמה העליונה, עשוי אובייקט Assistant להופיע על המסך עם האובייקט Balloon, התלוי בו מבחינה היררכית, או בלעדיו. הואיל והמסייע משתמש במיגוון רחב של דמויות והנפשות, הוא עשוי לבדר ולהציג מידע גם ללא הסברים. אם ברצונך לכלול הסברים, תוכל לתכנת אובייקטים מסוג Balloon כך שיופיעו בצמוד למסייע. בלונים עשויים לכלול הסברים, או אפילו לשמש כמנגנון פשוט להזנת נתונים. השתמש באוספים BalloonCheckBox ו-BalloonLabel עם לחצני פקודה כדי להפוך את המסייעים לאינטראקטיביים.



תרשים 8.3: השתמש באובייקט Assistant להצגה והנפשה של מסייע, ולהצגת בלון

מסייעים

מסייעים עשויים להביא תועלת רבה ביישומים על ידי הצגה חזותית של הפעולות השונות שהיישום מבצע. תוכל להוסיף נופך ייחודי ליישומיך המותאמים אישית על ידי שימוש עקבי במסייע מסוים, או על ידי שימוש באוסף של מסייעים שונים בנסיבות שונות. לדוגמה, השתמש במסייע **הגאון** (Genius) להצגת עזרה הקשורה בחישובים, ולעומת זאת, כאשר מדובר בעזרה בנושאים הקשורים במערכות מחשבים, השתמש במסייע הנראה כרובוט (**F1**). צוות הפיתוח שלך יוכל לקבוע לעצמו כללים בדבר הפעלת הנפשה מסוימת לכל פעולה או משימה האורכות מעל שתי שניות.

הצגה והנפשה של מסייע

תוכל לשלוט בנקל בתצוגה ובהנפשה של מסייע באמצעות שלוש השגרות הקצרות שנציג להלן: AssistantIdleOn, AssistantNotVisible ו-AssistantSearchOn. השיגרה IdleOn מכילה שורה אחת בלבד, המגדירה למאפיין Visible (גלוי) של האובייקט Assistant ערך True. הואיל ו- msoAnimationIdle הוא ברירת המחדל לסוג הנפשה (דמות ההנפשה אינה פעילה), השורה המגדירה את המאפיין Animation אינה נחוצה ולכן היא מופיעה בקוד כהערה. נוסף לכך, הנפשה בסיסית זאת היא חלק מספריית הליבה של Access, לכן תוכל להפעילה ללא הפניה אל הספריה Microsoft Office 9 Object Library. תוכל להפעיל גם את השיגרה AssistantNotVisible ללא הפניה לספריית האובייקטים של Office.

```
Sub AssistantIdleOn()

' Setting animation for idle is optional.
'    Assistant.Animation = msoAnimationIdle
   Assistant.Visible = True
End Sub

Sub AssistantNotVisible()
      Assistant.Visible = False
End Sub

Sub AssistantSearchOn()
      If Assistant.Visible = False Then Assistant.Visible = True
   Assistant.Animation = msoAnimationSearching
End Sub
```

כדי להחליף מצב 'לא פעילי' של דמות ההנפשה במצב אחר כלשהו, חובה להגדיר במסגרת השיגרה הן את המאפיין Animation של האובייקט, והן את המאפיין Visible. השיגרה AssistantSearchOn גורמת לאובייקט Assistant להציג את הנפשת החיפוש. שלא כמו הנפשות רבות אחרות, הנפשת החיפוש תחזור על עצמה עד שתקצה הגדרה חדשה למאפיין Animation. נוסף לכך, חובה לכלול בפרויקט הפניה לספריית האובייקטים של Office כדי שהנפשת החיפוש תופיע (ניתן לעשות זאת באופן ידני, באמצעות התפריט **Tools** (כלים) ב- Visual Basic Editor).

הפניה אוטומטית לספריית האובייקטים של Office

אחד החסרונות של השיגרה AssistantSearchOn הוא, שהשיגרה מניחה שיש הפניה לספריית האובייקטים של Office. אם אין הפניה כזאת, השיגרה נכשלת ללא כל סימן. המסייע מופיע, אולם הוא מציג הנפשה לא פעילה במקום הנפשת חיפוש. אחת הדרכים לפתור בעיה זאת היא לוודא שיש הפניה לספריית האובייקטים של Office, ולהוסיף הפניה אוטומטית במידת הצורך. אם תעשה כך, יוכל היישום להפעיל בבטחה כל הנפשה, ולמעשה כל מאפיין אחר של אובייקט Office משותף. השגרות AssistantSearchOn2 ו- ReferenceOffice שלהלן מציגות פתרון זה.

```
Sub AssistantSearchOn2()
   ReferenceOffice
   AssistantSearchOn
End Sub
```

```
Sub ReferenceOffice()
Dim ref1 As Reference
Dim blnOffice9In As Boolean, mso9Library As String
' Enumerate members of References collection to determine
' whether Office Object Library is already referenced.
    blnOffice9In = False
    For Each ref1 In References
        If ref1.name = "Office" Then
            blnOffice9In = True
        End If
    Next ref1

' If Office Object Library reference is missing, reference it.
    If blnOffice9In = False Then
        mso9Library = "C:\program files\Microsoft Office\Office\mso9.dll"
        Application.References.AddFromFile mso9Library
    End If

End Sub
```

השיגרה AssistantSearchOn2 זהה כמעט לחלוטין לשיגרה המקורית להפעלת הנפשת חיפוש. למעשה, היא אכן קוראת לשיגרה AssistantSearchOn, אולם היא מפעילה תחילה את ReferenceOffice. השיגרה ReferenceOffice מוודאת שקיימת בפרויקט הנוכחי הפניה לספריית האובייקטים של Office. היא פותחת במניית כל חברי האוסף References כדי לברר אם המאפיין Name באחד מהם הוא Office. אם כן, השיגרה מגדירה למשתנה בוליאני ערך True. אם לא, הגדרת המשתנה blnOffice9In נותרת False, לפי ברירת המחדל. המקטע השני של ReferenceOffice יוצר הפניה לספריית האובייקטים של Office כאשר ערך המשתנה הבוליאני הוא False. לאחר שווידאה שקיימת הפניה לספריית האובייקטים של Office, יכולה השיגרה AssistantSearchOn2 להפעיל את השיגרה AssistantSearchOn ולסיים בכך את הפעלת ההנפשה.

הצגת הנפשת חיפוש

ניתן להשתמש באובייקט Assistant להשלמת פעולת האובייקט FileSearch. כבר הזכרנו קודם כי בחלק מהמקרים, חיפוש הוא תהליך העשוי לארוך זמן-מה. במקרה כזה, רצוי להציג רמז כלשהו על המסך, כך שהמתבונן יידע שהיישום אכן עוסק בפעולה כלשהי. הנפשת החיפוש של המסייע מיטיבה לשרת מטרה זאת.

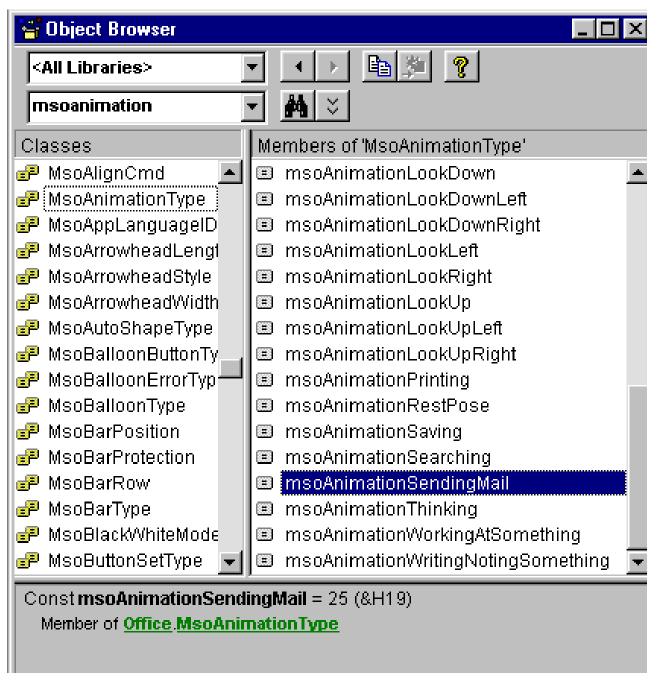
השיגרה הבאה מחפשת קבצי mdb. בכל רחבי כונן C ומחזירה את מניין הקבצים שאיתרה. באמצעות הנפשת חיפוש המוצגת מרגע שמתחיל החיפוש והחזרת ההנפשה הלא-פעילה מייד עם גמר החיפוש, המסייע מראה למשתמש מתי החיפוש בעיצומו ומתי הוא מסתיים. לשם כך נעשה שימוש בשתי קריאות לשגרות – אחת ממש לפני הפעלת השיטה Execute, והשנייה מייד לאחר סיומה. הואיל והאובייקטים FileSearch

ו- Assistant תלויים בספריית האובייקטים של Office, השיגרה מפעילה בשלב הראשון את השיגרה ReferenceOffice. אם לא קיימת הפניה לספריית האובייקטים של Office, השיגרה יוצרת אותה. בלא אמצעי זהירות זה, השיגרה תיכשל אם מסיבה כלשהי בוטלה ההפניה לספריית האובייקטים של Office.

```
Sub FileSearchAct()
' Reference the Office Object Library before using
' either the FileSearch or the Assistant objects.
   ReferenceOffice
' Search on C drive and its subfolders for *.mdb.
   With Application.FileSearch
' Start a new search.
      .NewSearch
' Set search criteria.
      .LookIn = "C:\"
      .SearchSubFolders = True
      .FileName = "*.mdb"
   End With
   With Application.FileSearch
' Execute the search.
' Turn searching assistant on first.
      AssistantSearchOn
      If .Execute() > 0 Then
         AssistantIdleOn
         MsgBox "There were " & .FoundFiles.Count & " file(s) found."
      Else
' If no files found, say so.
         MsgBox "There were no files found."
      End If
   End With
End Sub
```

בחירת סוג הנפשה

ספריית האובייקטים של Office מכילה מעל 30 סוגים שונים של הנפשות. תרשים 8.4 מציג קבוצת משנה של סוגי הנפשה כפי שהיא נראית דרך **Object Browser** (סורק האובייקטים). רוב ההנפשות, כמו למשל msoAnimationSendingMail ו- msoAnimationPrinting מבצעות מחזור הנפשה אחד וחוזרות למצב לא-פעיל. אחרות, כמו למשל msoAnimationSearching ו- msoAnimationThinking, חוזרות על עצמן עד שמופעלת הנפשה אחרת. בזכות IntelliSense, אין צורך לזכור את השמות הקבועים כדי להפנות להנפשות. פעמים רבות, עליך רק לבחור בהנפשה הרצויה מתוך רשימה.



תרשים 8.4: החבר msoAnimationType בספריית האובייקטים של Office מכיל מעל 30 קבועי הנפשה

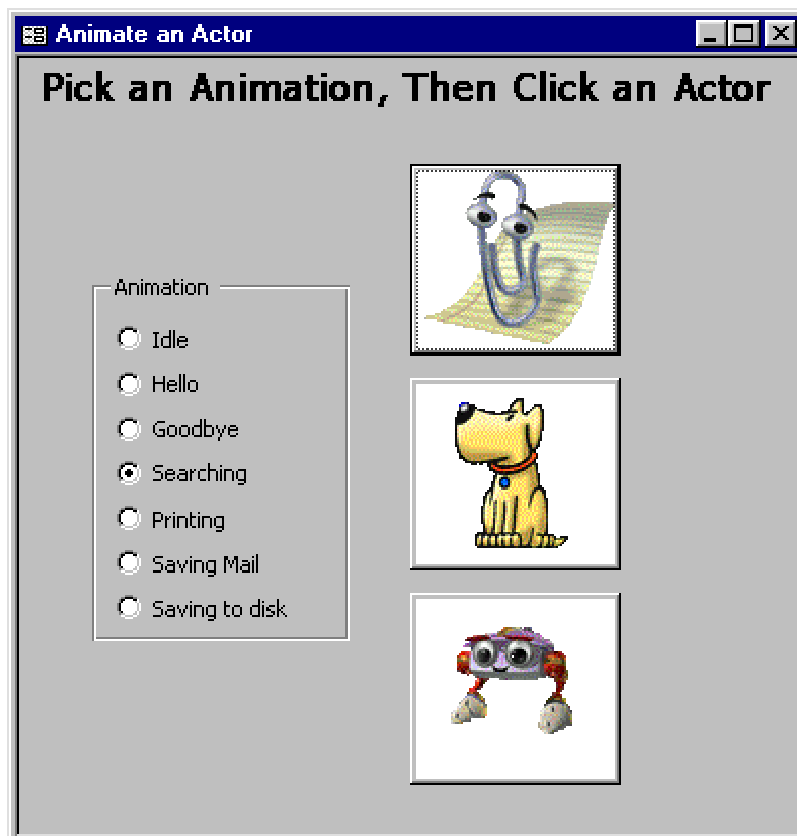
בחירת דמות למסייע

תוכנת Office 2000 משווקת עם שמונה דמויות שונות למסייע, אולם בהחלט ייתכן שחברת Microsoft תצרף דמויות נוספות בעתיד. הצגת הדמות הרצויה של המסייע נעשית על ידי קביעת שם הקובץ המתאים כהגדרת המאפיין FileName במסייע. הטבלה הבאה מציגה את שמות הדמויות השונות של המסייע, לצד שמות הקובץ המתאימים.

דמות המסייע	שם הקובץ
המהדק (Clippit)	Clippit.acs
לינקס (Links)	OffCat.acs
רוקי (Rocky)	Rocky.acs
סמל Office (Office Logo)	Logo.acs
הנקודה (The Dot)	Dot.acs
אמא אדמה (Mother Nature)	Mnature.acs
הגאון (The Genius)	Genius.acs
F1	F1.acs

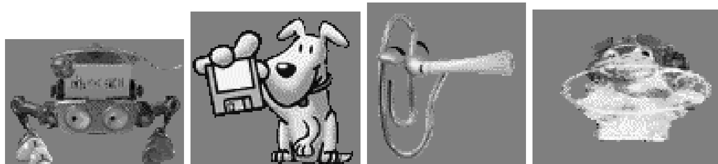
תצוגה מקדימה של הנפשות במסייע

תרשים 8.5 מציג טופס שבו מופעלות ההנפשות של המסייעים. ניתן להציג אותה הנפשה בכל מופע, או ליצור הבדלים בין מופע למופע. לדוגמה, קיימות לפחות שלוש גירסאות שונות של הנפשת 'להתראות' עבור הרובוט ששמו F1. הטופס מציג שבעה סוגי הנפשה עבור כל אחד משלושת המסייעים. השתמש בטופס לתצוגה מקדימה של הנפשות על ידי בחירה מתוך קבוצת אפשרויות ההנפשה, ולאחר מכן לחיצה על אחד מלחצני הפקודה של המסייעים. ניתוח של הקוד המפעיל את הטופס יראה לך כיצד לשלב הנפשות ולהחליף מסייעים ביישומיך.



תרשים 8.5: השתמש בטופס זה ליצירת תצוגה מקדימה של הנפשות המסייע.

תרשים 8.6 שלהלן מציג מבחר סוגים של מסייעים והנפשות שניתן ליצור בתצוגה מקדימה בעזרת הטופס שבתרשים 8.5. בקצה הימני, הרובוט F1 מבצע את הנפשת 'להתראות' על ידי היעלמות בתוך כדור אש. הדמות המהדק (Clippit) מגלגלת פיסת נייר ומשתמשת בה להצגת הנפשת החיפוש. בהנפשה 'שמירה בדיסקט', דמות הכלב רוקי (Rocky) מציגה לראווה דיסקט ולאחר מכן מניחה אותו לשמירה בתוך כיס בקולר. בהנפשת ההדפסה, הרובוט F1 הופך למדפסת.



תרשים 8.6: מבחר סוגי הנפשה שמבצעים שלושה מסייעים שונים. הנפשות אלה הופקו בעזרת הטופס המוצג בתרשים 8.5.

דוגמת הקוד הבאה מציגה את הקוד שמאחורי הטופס המוצג בתרשים 8.5. אזור ההצהרה שבראש המודול מכיל את הגדרות זמן-העיצוב של לחצני הפקודה המראים את סוגי המסייעים, ואת הלחצנים של סוגי ההנפשה השונים בקבוצת האפשרויות. שים לב ששמות הקבצים המתאימים למסייעים הנראים על פני הלחצנים מאוחסנים במאפיין Tag (תג) שלהם. לכל לחצן גם שגרת אירוע פשוטה לאירוע הלחיצה המתאים לו. לכל אחד מהלחצנים שבקבוצת האפשרויות ערך המציין קבוע msoAnimationType מסוים. השתמש בסורק האובייקטים המוצג בתרשים 8.4 כדי לברר את הערך הנומרי של קבוע נתון (לדוגמה, הערך 25 מייצג את שם הקבוע msoAnimationsSendingMail). כאשר הטופס נטען, הוא מגדיר סדרה של מאפיינים כדי להבטיח שהמסייע יהיה מוכן בעת שהמשתמש בוחר בלחצן מתוך קבוצת האפשרויות ולוחץ על אחד מלחצני הפקודה.

```
Option Compare Database
'
' Design-time Command button settings
'    Name: cmdClippit , cmdRocky, cmdF1
'    Picture: c:\My Documents\My Pictures\clippit.bmp,
'        or rocky.bmp, or F1.bmp
'    Picture Type: Embedded
'    Tag: clippit.acs, or rocky.acs, or F1.acs
'    On Click: [Event Procedure]
'
' Design-time Option button settings
'    Idle button's Option Value = 1
'    Hello button's Option Value = 2
'    Goodbye button's Option Value = 3
'    Searching button's Option Value = 13
'    Printing button's Option Value = 18
'    Saving Mail button's Option Value = 25
'    Saving to disk button's Option Value = 112

Private Sub Form_Load()
    With Assistant
        .On = True
        .Sounds = True
        .Visible = True
    End With
End Sub
```

```
Private Sub cmdClippit_Click()
   AnimateActor Me.Controls("cmdClippit").Tag
End Sub

Private Sub cmdRocky_Click()
   AnimateActor Me.Controls("cmdRocky").Tag
End Sub

Private Sub cmdF1_Click()
   AnimateActor Me.Controls("cmdF1").Tag
End Sub

Sub AnimateActor(Fname As String)
   With Assistant
      .FileName = Fname
      .Animation = _
         Me.Controls("optAnimation")
      .Visible = True
   End With
End Sub
```

לחצני הפקודה הם גורם מרכזי בתהליך. כאשר לוחצים על לחצן פקודה, שגרת האירוע מעבירה את ערך התג שלה לשיגרה AnimateActor. שיגרה זאת משתמשת בארגומנט המועבר שלה להגדרת המאפיין FileName של המסייע. לאחר מכן היא מגדירה למאפיין Visible של המסייע ערך True ולמאפיין Animation את ערך קבוצת האפשרויות. ערך קבוצת האפשרויות הוא פונקציה של הלחצן שבחרת. ערך ברירת המחדל של הפקד הוא 1, למקרה שהמשתמש פסח על בחירת הלחצן.

בלונים

באמצעות בלונים, תוכל להציג טקסט או גרפיקה, או לאסוף משוב מהמשתמש. ניתן להציג בלונים כתיבת דו-שיח מודאלית, נטולת מודאליות, או נסגרת-אוטומטית. המאפיין Mode של הבלון קובע את סוג הבלון. ההגדרה msoModeAutoDown סוגרת את הבלון בעקבות לחיצה שלוחץ המשתמש בכל נקודה במסך. ההגדרה נטולת המודאליות (msoModeModeless) והמודאלית (msoModeModal) שכיחות יותר. ההגדרה נטולת המודאליות גורמת לבלון להישאר פתוח בשעה שהמשתמש מבצע פעולות מחוץ לגבולותיו. ההגדרה המודאלית כופה תגובה בטרם יוכל המשתמש להמשיך בכל פעולה אחרת. ערך ברירת המחדל למאפיין Mode הוא msoModeModal.

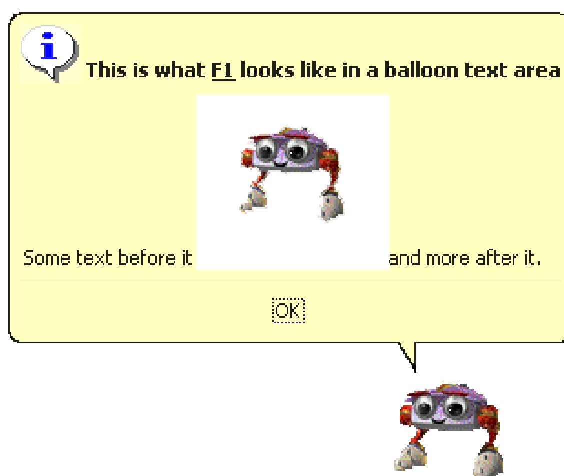
השתמש במאפיין NewBalloon של האובייקט Assistant ליצירת אובייקט Balloon. לכל בלון אזורים נפרדים לכותרת, טקסט, תווית, תיבת סימון ולחצנים. תוכל לאכלס אזורים אלה בעזרת הגדרות המאפיין המתאימות, או באמצעות אובייקטים היררכיים כמו BalloonCheckBoxes ו-BalloonLabels. הקצאת טקסט לתיבות הסימון ולתוויות

נעשית באמצעות המאפיין Text שלהן. תוכל לבצע התאמה אישית נוספת בבלון בעזרת המאפיין Icon שלו. ישנם שישה סמלים שונים לשישה סוגים שונים של מאפייני הודעה, כמו למשל התראה, שאלה, מידע ועצה.

גרפיקה בבלונים

השיגרה הבאה מציגה בלון עם כותרת, טקסט וסמל. כמו כן נעשה בה שימוש בלחצן ברירת המחדל (**OK**). מחרוזות עם עיצוב מילוט (escape) שנמצאות בטקסט המאפיין Heading (כותרת) מציינות את תחילתו ({ul 1}) וסופו ({ul 0}) של טקסט המעוצב בקו-תחתון באזור הכותרת (ניתן להשתמש במחרוזות עם עיצוב מילוט גם לצורך עיצוב טקסט בקו-תחתון באזור הטקסט). הגדרת המאפיין Text כוללת תמונת מפת-סיביות באזור הטקסט. שים לב שניתן לבצע גלישת טקסט מסביב לתמונה. הגדרת המאפיין Icon (סמל) מציינת שהבלון מכיל מידע. לבסוף, השיטה Show פותחת את המסייע ואת הבלון המקושר אליו. תרשים 8.7 מציג את המסייע והבלון המופיעים עם הפעלת השיגרה balloonTextImageIcon.

```
Sub balloonTextImageIcon()
    With Assistant.NewBalloon
        .Heading = "This is what {ul 1}F1{ul 0} looks like in a balloon text area"
        .Text = " Some text before it " & _
            "{bmp ""C:\My Documents\My Pictures\F1.bmp""} " & _
            "and more after it."
        .Icon = msoIconAlertInfo
        .Show
    End With
End Sub
```



תרשים 8.7: ניתן להוסיף גרפיקה לבלון, ולהגדיר את מהות התכולה באמצעות הגדרות המאפיין Icon (כמו למשל הגדרת הסמל AlertInfo המשמשת בבלון שבתרשים זה).

תוויות של בלונים

שתי השגרות שלהלן מכינות בלון עם תוויות, אך ללא לחצנים. השיגרה setLabelCount מעבירה ארגומנט לשיגרה balloonHeadTextLabel. עבור ארגומנטים חוקיים מ-1 ועד 5, כולל, השיגרה המופעלת מציגה מסייע עם בלון המכיל תוויות כמספר הערך שהועבר לשיגרה. כל תווית מופיעה עם מחרוזת טקסט פשוטה המכריזה על מספר התווית. בעיצוב זה אין צורך בלחצנים, הואיל ולחיצה על אחת התוויות גורמת לסגירת הבלון. השיטה Show פותחת את הבלון ומעבירה את ערך התווית שהמשתמש לחץ עליה למשתנה i שהוא מסוג Integer. תיבת הודעה מציגה את מספר התווית שהמשתמש לחץ עליה כדי לסגור את הבלון.

```
Sub setLabelCount()
    balloonHeadTextLabel 5
End Sub

Sub balloonHeadTextLabel(LabelCount As Integer)
On Error GoTo LabelTrap
Dim b1 As balloon, i As Integer

' Check for 0 or negative label count.
    If LabelCount <= 0 Then
        Err.Raise 9
    End If
    Set b1 = Assistant.NewBalloon

' Create balloon with specified number of labels.
    With b1
        .Heading = "This is my heading"
        .Text = "Select one of these things:"
        For i = 1 To LabelCount
            .Labels(i).Text = "Label " & i
        Next i
        .Button = msoButtonSetNone
        i = .Show
    End With

' Confirm label that user clicked.
    MsgBox i, vbInformation, "Programming Microsoft Access 2000"

LabelExit:
    Exit Sub
```

```
LabelTrap:
    If Err.Number = 9 Then
        MsgBox "Number of labels more than 5 or less than " & _
            "1. Retry with another number.", vbInformation, _
            "Programming Microsoft Access 2000"
    Else
        Debug.Print Err.Number
        MsgBox "Unanticipated error. Error number = " & _
            Err.Number & vbCrLf & "Error Description: " & _
            Err.Description, vbInformation, _
            "Programming Microsoft Access 2000"
    End If

    Resume LabelExit

End Sub
```

גילוי ארגומנטים לא חוקיים למספר התוויות נעשה באמצעות לוגיקה מוכללת ומותאמת אישית ללכידת שגיאות. אם הארגומנט המועבר גדול מ-5, הלולאה For...Next המקצה ערכי תווית תיכשל בניסיונה להקצות טקסט לתווית השישית. שגיאה זאת (שגיאה מספר 9) נובעת מכך שבלונים יכולים להציג חמש תוויות לכל היותר. ארגומנט בעל ערך אפס או ערך שלילי המועבר לשיגרה לא יחולל שגיאה מסוג זה, אולם העברתו לא תאכלס את הבלון בתוויות. לכן, השיגרה balloonHeadTextLabel לוכדת ערכים אלה בעזרת משפט If...Then, ויוצרת שגיאה מותאמת אישית שמספרה 9. המלכודת שמכילה השיגרה לשגיאה זאת מורה למשתמש להזין ערכים בטווח שבין 1-5 בלבד.

איסוף קלט מהמשתמש בעזרת בלונים

האוסף BalloonCheckBoxes מאפשר הצגה של קבוצת אפשרויות ואיסוף תגובות מהמשתמש. גם כאן הנך מוגבל לחמישה פקדים – במקרה זה הפקדים הם תיבות סימון. שלא כמו פקדי תווית, פקדי תיבת סימון אינם סוגרים את הבלון עם בחירתם. לכן, אם ציינת בלון עם תיבות סימון, עליך להקצות ערכת לחצנים לסגירת הבלון.

שתי השגרות הבאות מציגות מסגרת כללית להצגת פקדי תיבת סימון בתוך בלון. הן אינן כוללות לכידת שגיאות, אולם בפועל, עליך לכלול לוגיקה לטיפול בשגיאות כמו זאת שבדוגמה הקודמת. השיגרה setCheckCount מעבירה ארגומנט בטווח שבין 1 ל-5 כדי לציין כמה תיבות סימון יש לכלול בבלון. השיגרה השנייה נפתחת בהגדרת הפניה b1, לבלון חדש. היא מקצה למאפיין Button של האובייקט Balloon את ההגדרה MsoButtonSetOkCancel כדי לכלול בבלון לחצני אישור (OK) וביטול (Cancel). דבר זה מאפשר סגירה של הבלון בשתי דרכים. לאחר הגדרת הערכים עבור הכותרת והטקסט בבלון, השיגרה מקצה טקסט לכל אחת מתיבות הסימון. ערך ההחזרה מהשיטה Show במקרה זה מציין את הלחצן שעליו לחץ המשתמש.

```
Sub setCheckCount()
    balloonHeadTextCheck 5
End Sub

Sub balloonHeadTextCheck(CheckCount As Integer)
Dim b1 As balloon, i As Integer
Dim strChecks As String, iChecks As Integer

' Set reference to the balloon.
    Set b1 = Assistant.NewBalloon

' Assign text to check box controls.
    With b1
        .Button = msoButtonSetOkCancel
        .Heading = "Here's the Heading"
        .Text = "This is some text in the balloon."
        For i = 1 To CheckCount
            .Checkboxes(i).Text = "Check box # " & i
        Next
        i = .Show
    End With

' Did user click Cancel button?
    If i = msoBalloonButtonCancel Then
        MsgBox "You cancelled the balloon.", vbInformation, _
            "Programming Microsoft Access 2000"
        Exit Sub
    End If

' Record individual boxes checked and count of all checked boxes.
    For i = 1 To CheckCount
        If b1.Checkboxes(i).Checked = True Then
            If strCheck = "" Then
                strCheck = CStr(i)
            Else
                strCheck = strCheck & ", " & CStr(i)
            End If
            iChecks = iChecks + 1
        End If
    Next i
```

```
' Present a message box with the results.
    If iChecks = 0 Then
        MsgBox "No boxes checked.", vbInformation, _
            "Programming Microsoft Access 2000"
    ElseIf iChecks = 1 Then
        MsgBox "You checked box " & strCheck & ".", _
            vbInformation, "Programming Microsoft Access 2000"
    Else
        MsgBox "You checked " & iChecks & " boxes. " & _
            "These were boxes: " & strCheck & ".", _
            vbInformation, "Programming Microsoft Access 2000"
    End If
End Sub
```

שארית השיגרה balloonHeadTextCheck מעבדת את התגובה לבלון. תחילה, השיגרה בודקת את ערך ההחזרה מהשיטה Show. אם ערך זה שווה לקבוע msoBalloonButtonCancel, המשתמש לחץ על הלחצן ביטול. לאחר הצגת ההודעה המתאימה על כך, השיגרה מסתיימת.

הואיל והדרך היחידה לסגור את הבלון היא ללחוץ על הלחצן אישור או ביטול, אם המשתמש לא לחץ על ביטול, מן הסתם לחץ על אישור. קטע הקוד הבא מבצע שתי משימות: הוא מפתח מחרוזת טקסט שמציינת תיבות מסומנות, ולאחר מכן מונה את התיבות הללו. קטע הבלוק האחרון בשיגרה משתמש במחרוזת ובתוצאת הספירה להכנת תיבת הודעה שבה ייאמר, בהתאם לתוצאה, שאף תיבה לא סומנה, שסומנה תיבה אחת בלבד, או שסומנו מספר תיבות.

בלונים לא מודאליים

הדוגמה הבאה מעבדת בלונים נטולי מודאליות, שנשארים פתוחים בשעה שהמשתמש מבצע פעולה כלשהי בחלק אחר של היישום. כל הדוגמאות הקודמות עסקו בבלונים מודאליים. אחת התכונות המותאמות אישית לעבודה עם בלונים נטולי מודאליות היא המאפיין Callback (התקשרות חזרה). מאפיין זה מקבל כערך את שמה של שיגרה אחרת המבצעת פעולה כלשהי עם התגובה לבלון. על השיגרה Callback לבלונים נטולי מודאליות להכיל שלושה פרמטרים בסדר קבוע. הפרמטר הראשון כולל משתנה המייצג את הפניית הבלון. הפרמטר השני כולל משתנה לציון הפקד שהמשתמש לחץ עליו. הפרמטר השלישי משתמש במשתנה ארוך המזהה את המאפיין Private של הבלון המפעיל את שגרת ההתקשרות חזרה. בבלון נטול מודאליות, חשוב במיוחד ליצור פקד לסגירה. אם אינך מציג פקד כזה, הבלון עלול להישאר פתוח ללא הגבלת זמן.

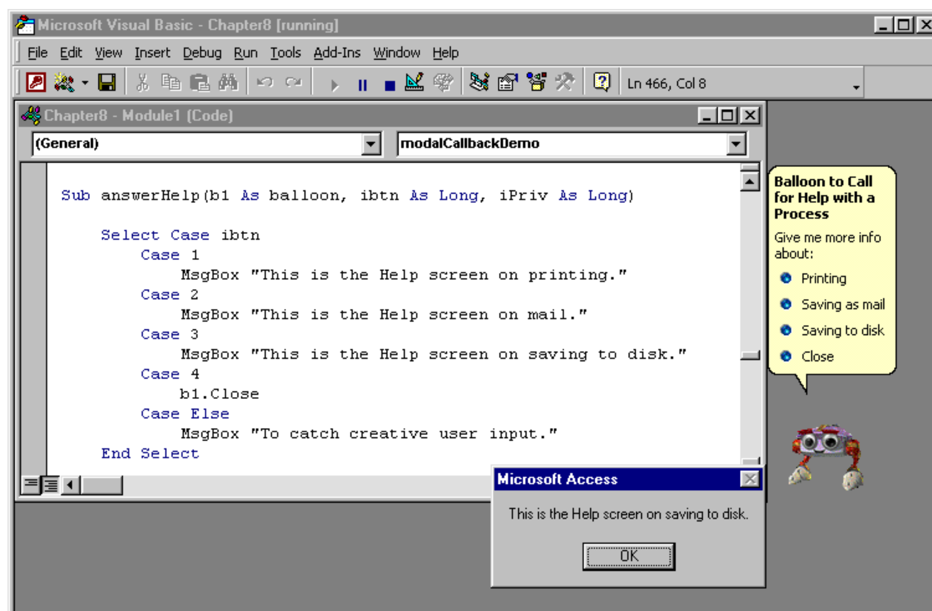
השיגרה שלהלן מציגה בלון עם ארבע תוויות. שים לב, שהגדרת המאפיין Mode של הבלון היא msoModeModeless. הגדרה זאת מחייבת הגדרה של המאפיין Callback. ללא הגדרה כזאת, השיגרה תחולל שגיאה. שים לב גם, שאחת התוויות מציינת

במפורש אפשרות סגירה. ניתן להציג אפשרות סגירה גם באמצעות הגדרה של המאפיין Button. לחיצה על תוויות לא תגרום באופן אוטומטי לסגירת הבלון; היא תפעיל את שגרת ההתקשרות חזרה הקרויה answerHelp. שיגרה זאת מוצגת בתרשים 8.8, להלן.

```
Sub modalCallbackDemo()
Dim b1 As balloon
    Set b1 = Assistant.NewBalloon

    With b1
        .Heading = "Balloon to Call for Help with a Process"
        .Text = "Give me more info about:"
        .Labels(1).Text = "Printing"
        .Labels(2).Text = "Saving as mail"
        .Labels(3).Text = "Saving to disk"
        .Button = msoButtonSetNone
        .Mode = msoModeModeless
        .Callback = "answerHelp"
        .Show
    End With

End Sub
```

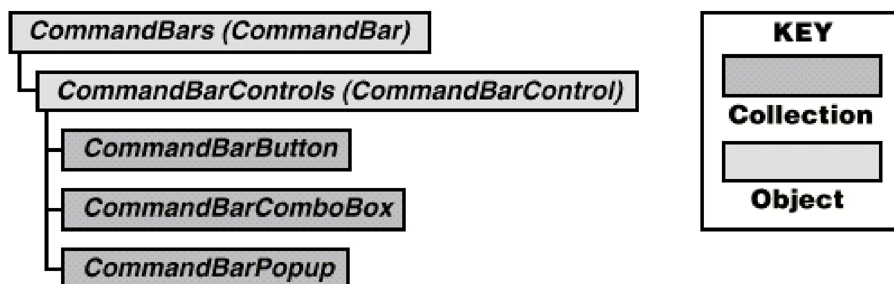


תרשים 8.8: בלון נטול מודאליות ושגרת ההתקשרות חזרה שלו.

השיגרה answerHelp מציגה תיבת הודעה שתכולתה נקבעת על ידי התווית שהמשתמש לחץ עליה בבלון. המשפט Select Case המעבד את התגובה קורא את התווית שבחר המשתמש מתוך הפרמטר השני המועבר לשגרת ההתקשרות חזרה. לאחר שהמשתמש סוגר את תיבת ההודעה, הבלון נשאר פתוח. ככלל, יוכל המשתמש להמשיך ולנווט בתוך טופס, בעוד הבלון נטול המודאליות מציג עזרה על אופן התגובה לשדות השונים בטופס. אם המשתמש ילחץ על הלחצן הרביעי, השיגרה תשתמש בהפניה b1 לבלון להפעלת השיטה Close.

האובייקט CommandBar

מבנה האובייקט CommandBar (המוצג בתרשים 8.9) עשיר ביותר – הוא כולל סרגלי כלים מוכללים ומותאמים אישית כאחד. **Command Bar** (סרגל פקודות) הוא מונח כללי המציין שורת תפריטים, סרגל כלים או שורת תפריטים מוקפצת. פקדי CommandBar מאפשרים למשתמש לפעול עם יישום נתון, באמצעות ממשק לסרגלי הפקודות של היישום. פקדי CommandBar נחלקים לשלוש מחלקות אובייקטים רחבות: CommandBarButton, CommandBarComboBox ו- CommandBarPopup.



תרשים 8.9: השתמש במבנה האובייקט CommandBar להתאמה אישית של סרגלי פקודות מוכללים, וליצירת סרגלי פקודות אישיים

ספירת האלמנטים בסרגלי פקודות

ספירה של האלמנטים בסרגל הפקודות חיונית בעבודה עם סרגלי פקודות. הספירה היא כלי חשוב לבירור ההירכיה של האובייקט CommandBar. המידע שיתברר לך באמצעות הדוגמאות הבאות יסייע בידך לערוך שינויים בסרגלי פקודות מוכללים, ובפיתוח סרגלי פקודות מותאמים אישית.

השיגרה הקצרה שלהלן מפיקה את מניין סרגלי הפקודות ביישום. אם אין ביישום סרגלי פקודות מותאמים אישית, השיגרה מפיקה את מניין סרגלי הפקודות המוכללים, 140 במספר. מספר זה גבוה יותר כאשר יישום מכיל גם סרגלי פקודות מותאמים אישית.

```
Sub countCommandBars()
   MsgBox "There are " & CommandBars.Count & _
      " bars in the CommandBars collection."
End Sub
```

לרשותך שלושה סוגים של סרגלי פקודות. ספריית האובייקטים של Office מכילה את קבועי msoBarType להפניה אל סרגלי פקודות מסוגים אלה: msoBarTypeNormal, msoBarTypeMenuBar ו- msoBarTypePopup. כמו כן ניתן להבחין בין סרגלי פקודות מוכללים לסרגלי פקודות מותאמים אישית. השיגרה הבאה מונה את סרגלי הפקודות המוכללים, לפי סוג.

```
Sub builtinCommandBarCount()
Dim cbr1 As CommandBar
Dim iMbars As Integer, iTbars As Integer
Dim iPbars As Integer, ibuiltin As Integer
   For Each cbr1 In CommandBars
      If cbr1.BuiltIn Then
         ibuiltin = ibuiltin + 1
         If cbr1.Type = msoBarTypeMenuBar Then
            iMbars = iMbars + 1
         ElseIf cbr1.Type = msoBarTypeNormal Then
            iTbars = iTbars + 1
         Else
            iPbars = iPbars + 1
         End If
      End If
   Next
   MsgBox "There are " & ibuiltin & " command bars. " & _
      IMBars & "is a menu bar, " & iTbars & " are toolbars, and " & _
      iPbars & " are popup bars."
End Sub
```

רשימת סרגלי הפקודות הגלויים לעין

מכיון שקיימים 140 סרגלי פקודות מוכללים, מן הסתם לא תרצה למנות את כולם בדרך כלל. עם זאת, עשויה להיות חשיבות לקבוצות משנה שונות ביישום נתון. לדוגמה, היישום עשוי להזדקק לנתון, אלו סרגלי פקודות נראים לעין. השיגרה enumerateVisibleCommandBars כותבת שורה לחלון Immediate (מיידי) עבור כל סרגל פקודות נראה לעין. השורות מציגות שלושה מאפיינים לכל סרגל פקודות – שם סרגל הפקודות, סוגו ומספר הפקדים שהוא מכיל. זוג פונקציות Immediate If מקוננות מפענח את המאפיין Type. במקום למנות את הפקדים בסרגלי הפקודות כדי להגיע לספירה, השיגרה פשוט מציגה את המאפיין Count של סרגל הפקודות.

```
Sub enumerateVisibleCommandBars()
Dim cbr1 As CommandBar
   For Each cbr1 In CommandBars
      If cbr1.Visible = True Then
         Debug.Print cbr1.name, _
            (IIf(cbr1.Type = msoBarTypeNormal,"toolbar", _
            IIf(cbr1.Type = msoBarTypeMenuBar, "menu bar", "popup bar"))), _
            cbr1.Controls.Count
      End If
   Next cbr1
End Sub
```

קל למדי להרחיב את קטע הקוד שלעיל כך שימנה את הפקדים הנפרדים בכל סרגל פקודות נראה לעין. לכל סרגל פקודות אוסף Controls, והאלמנטים של אוסף זה הם אובייקטי CommandBarControl. השיגרה הבאה מחילה אובייקט CommandBar ואובייקט CommandBarControl תוך הצגת רשימת הכיתובים של הפקדים בכל סרגלי הפקודות הנראים לעין.

```
Sub enumerateControlCaptions()
Dim cbr1 As CommandBar
Dim ctl1 As CommandBarControl
   For Each cbr1 In CommandBars
      If cbr1.Visible = True Then
         Debug.Print "Command bar name: " & cbr1.name & _
            " and control count: "; cbr1.Controls.Count
            For Each ctl1 In cbr1.Controls
               Debug.Print cbr1.name, ctl1.Caption
            Next ctl1
      End If
   Next cbr1
End Sub
```

הצגת פקודות תפריט

לבסוף, ייתכן שתידרש לך רשימה של הפקודות הנפרדות בתפריט כלשהו. במקרה זה יש לטפל בתפריט כבסרגל פקודות, כך שהפקודות ייחשפו כפקדים. תוכל לקבוע את השם עבור סרגל פקודות המציג תפריט מסוים באמצעות השיגרה enumerateControlCaptions (או וריאציה שלה). שתי השגרות שלהלן יוצרות לולאה העוברת על הפקדים של תפריט נתון. השיגרה הראשונה מעבירה שם של סרגל פקודות לשיגרה השנייה, שעוברת בלולאה על פני הפקדים של אותו סרגל פקודות.

```
Sub listCommands()
        enumerateCommandsOnMenu ("Help")
End Sub

Sub enumerateCommandsOnMenu(menuName)
Dim cbr1 As CommandBar
Dim ctl1 As CommandBarControl

' Set a reference to a command bar.
    Set cbr1 = CommandBars(menuName)
' Loop through the controls for that command bar.
    For Each ctl1 In cbr1.Controls
        Debug.Print ctl1.Caption
    Next ctl1
End Sub
```

טיפול בסרגלי פקודות מוכללים

ניתן לשנות סרגלי פקודות מוכללים במספר דרכים. בסעיפים הבאים מוצגות כמה מהן.

שינוי מצב הזמינות של סרגלי הפקודות ופקדיהם

באפשרותך להפוך סרגלי פקודות לבלתי זמינים, ולאחר מכן להשיב אותם למצב זמין. שתי השגרות הבאות מבטלות את הזמינות של שורת התפריטים המוכללת (המכונה Menu Bar), ולאחר מכן מחזירות אותה למצב זמין. כדי לבטל את זמינותו של סרגל פקודות זה בטופס, עליך להגדיר למאפיין Enable שלו ערך False בתוך שיגרה של אירוע טופס. יישומיך יוכלו להתנות את ביטול הזמינות של סרגל פקודות בגורמים שונים, כמו למשל זיהוי משתמש.

```
Sub disableMenuBar()
Dim cbr1 As CommandBar

    For Each cbr1 In CommandBars
        If cbr1.name = "Menu Bar" Then
            cbr1.Enabled = False
        End If
    Next cbr1

End Sub
```

```
Sub enableMenuBar()
Dim cbr1 As CommandBar

   For Each cbr1 In CommandBars
      If cbr1.name = "Menu Bar" Then
         cbr1.Enabled = True
      End If
   Next cbr1

End Sub
```

כמו כן ניתן להפוך לבלתי זמינות פקודות מסוימות בשורת התפריטים או בסרגל כלים. השיגרה הראשונה מתוך שתי השגרות הבאות הופכת לבלתי זמינה את הפקודה **תצוגה** (View) שבשורת התפריטים ובסרגל הכלים **תצוגת טופס** (Form View). פעולה זאת מסייעת בשמירה על עיצוב הטופס, על ידי ביטול שני נתיבים מוכרים למעבר בין תצוגת **טופס** (Form) לתצוגת **עיצוב** (Design). נוסף לנטרול הפקד **תצוגה** (View), מגינה השיגרה הראשונה על השינוי על ידי קביעת ההגדרה msoBarNoCustomize למאפיין Protection של סרגל הפקודות. הגדרה זאת הופכת לבלתי זמין (מוצג באפור) את הלחצן **איפוס** (Reset) בתיבת הדו-שיח **התאמה אישית** (Custom) של סרגלי הפקודות **שורת תפריטים** (Menu Bar) **ו**תצוגת טופס**. השיגרה השנייה מחזירה את הפקודות בשני סרגלי הפקודות למצב זמין. שים לב כי שמות הפקודות מופיעים בקוד כשמות הפקדים בממשק Access (למשל, בממשק עברי שם הפקודה בקוד שלהלן הוא "תצוגה", ובממשק אנגלי שמה "View").

```
Sub disableViewMenuAndControl()
Dim ctl1 As CommandBarControl
' Disable and protect View Menu.
   Set ctl1 = CommandBars("Menu Bar").Controls("תצוגה")
   ctl1.Enabled = False
   CommandBars("Menu Bar").Protection = msoBarNoCustomize
' Disable and protect View Control.
   Set ctl1 = CommandBars("Form View").Controls("תצוגה")
   ctl1.Enabled = False
   CommandBars("Form View").Protection = msoBarNoCustomize
End Sub

Sub enableViewMenuAndControl()
Dim ctl1 As CommandBarControl
' Enable View Menu.
   Set ctl1 = CommandBars("Menu Bar").Controls("תצוגה")
   ctl1.Enabled = True
' Enable View Control.
   Set ctl1 = CommandBars("Form View").Controls("תצוגה")
   ctl1.Enabled = True
End Sub
```

הצגת סרגלי פקודות בלתי נראים

פעולה נוספת, פשוטה אך חשובה, שניתן לבצע היא חשיפת תפריט מוכלל שבדרך כלל אינו מוצג. השיגרה הבאה מציגה את השם, הסוג ומספר הפקדים בכל סרגל פקודות נראה לעין. אם סרגל הכלים **אינטרנט** (Web) אינו נראה לעין, השיגרה מגדירה את המאפיין שלו Visible (גלוי) ומשאיר רישום של פעולה זו בחלון **Immediate** על ידי הדפסת השם, הסוג ומניין הפקדים של סרגל הכלים. תוכל להעלים את סרגל הכלים **אינטרנט** שוב על ידי הקצאת ערך False למאפיין Visible שלו.

```
Sub showWebBar()
Dim cbr1 As CommandBar
    For Each cbr1 In CommandBars
        If cbr1.Visible = True Then
            Debug.Print cbr1.name, cbr1.Type, cbr1.Controls.Count
        ElseIf cbr1.name = "Web" Then
            cbr1.Visible = True
            Debug.Print cbr1.name, cbr1.Type, cbr1.Controls.Count
        End If
    Next cbr1
End Sub
```

הוספת פקודות לסרגלי פקודות מוכללים

מלבד טיפול בחברים מוכללים באוסף CommandBars, באפשרותך להוסיף פקודות מותאמות אישית לכל סרגל כלים מוכלל. דרך פשוטה לעשות זאת היא להוסיף אובייקט CommandBarButton. עליך לדעת במדויק את שמו של סרגל פקודות כדי שתוכל להוסיף בו לחצן חדש בעזרת השיטה Add (זכור שבאפשרותך להפעיל את השיגרה enumerateControlCaptions כדי ליצור רשימה של שמות סרגלי הפקודות). לאחר הוספת הלחצן, הגדר את המאפיינים עבור אובייקט CommandBarButton החדש כך שיצביע אל שיגרה או פונקציה מותאמת אישית.

השיגרה newMenuItem שלהלן ושלוש השגרות הנלוות אליה מוסיפות פריטים חדשים לתפריט. השיגרה newMenuItem מוסיפה אובייקטי CommandBarButton בסופו של סרגל פקודות בשם Tools (כלים). שלוש השגרות הנלוות מאפשרות למשתמש לציין איזו דמות ילבש המסייע – המהדק (Clippit), רוקי (Rocky) או F1. אובייקטי CommandBarButton החדשים מאפשרים למשתמש להפעיל את השגרות השולטות בדמות המסייע שתוצג.

```
Sub newMenuItem()
Dim newItem As CommandBarButton
' Set reference to new control on the Tools command bar.
   Set newItem = CommandBars("Tools").Controls.Add(Type:=msoControlButton)
```

```
' Start new group with command to invoke showClippit.
   With newItem
        .BeginGroup = True
        .Caption = "Show Clippit"
        .OnAction = "showClippit"
   End With

' Set reference to new control on the Tools command bar.
   Set newItem = CommandBars("Tools").Controls.Add(Type:=msoControlButton)
' Assign command to invoke showRocky.
   With newItem
        .Caption = "Show Rocky"
        .OnAction = "showRocky"
   End With

' Set reference to new control on the Tools command bar.
   Set newItem = CommandBars("Tools").Controls.Add(Type:=msoControlButton)
' Assign command to invoke showRocky.
   With newItem
        .Caption = "Show F1"
        .OnAction = "showF1"
   End With
End Sub

Sub showRocky()
   With Assistant
        .Visible = True
        .FileName = "Rocky.acs"
        .On = True
   End With
End Sub

Sub showClippit()
   With Assistant
        .Visible = True
        .FileName = "Clippit.acs"
        .On = True
   End With
End Sub

Sub showF1()
   With Assistant
        .Visible = True
        .FileName = "F1.acs"
        .On = True
   End With
End Sub
```

השתמש בשיטה Add עבור האוסף Controls של סרגל פקודות כדי להוסיף פקד חדש בתפריט מוכלל. שיטה זאת מקבלת מספר ארגומנטים, לרבות הפרמטר Type. נוסף לפקד הלחצן (msoControlButton) שבדוגמה, תוכל לציין תיבת טקסט פשוטה (msoControlEdit), תיבה משולבת (msoControlComboBox), ועוד. לפי ברירת המחדל, השיטה Add מוסיפה את הפקד החדש בסופו של סרגל הפקודות, אולם ניתן לעקוף תכונה זאת ולהציג את הפקד במקום אחר בסרגל הפקודות. פרמטר נוסף, ID, מפשט את ההוספה של פקודות מוכללות מתפריטים אחרים לסרגל הפקודות המותאם אישית.

לאחר הוספת פקד לסרגל פקודות מוכלל, ניתן לקשור אותו לתפקוד מותאם אישית בעזרת המאפיין OnAction. קבע כערך המאפיין את שמה של שיגרה שאתה רוצה להפעיל באמצעות הפקד החדש. באמצעות המאפיין Caption של הפקד תוכל בנקל לתת תווית לפקד החדש. השתמש בשיטות CopyFace ו- PasteFace כדי לסמן את הפקדים המותאמים אישית. כאשר המאפיין BeginGroup מקבל ערך True, יופיע על סרגל הפקודות פקד, כשלפניו קו מפריד. בדוגמה, מאפיין זה מקבל ערך True עבור הראשון מבין שלושת הפקדים המותאמים אישית, אולם ערך ברירת המחדל False נותר בעינו עבור שני הפקדים האחרים.

שיחזור סרגלי פקודות

בתהליך הליטוש של יישומיך המותאמים אישית, ייתכן שתרצה להסיר פקדים מותאמים אישית שהוספת בתפריטים מוכללים. עשה זאת באמצעות השיטה Reset. השיגרה הבאה מסירה את כל הפקדים המותאמים אישית מסרגל הפקודות Tools.

```
Sub removeMenuItem()
   CommandBars("Tools").Reset
End Sub
```

יצירת סרגלי פקודות מותאמים אישית

יצירת סרגל פקודות מותאם אישית נעשית בשלושה שלבים לפחות:

1. הוספת סרגל פקודות חדש ליישום. הסרגל החדש יהיה ריק בתחילה.
2. הצבת פקדים בסרגל הפקודות. פעולה זאת דומה להוספת פקדים בסרגל פקודות מוכלל.
3. הגדרת ערך True למאפיין Visible של סרגל הפקודות אם ברצונך להציגו. כמו כן תוכל לאפשר למשתמשים לחשוף את סרגל הפקודות המותאם אישית שלך באמצעות התכונות הרגילות (כמו למשל תיבת הדו-שיח **התאמה אישית** (Customize)).

שתי השגרות הבאות יוצרות סרגל פקודות מותאם, ובו פקד לחצן אחד הגורם להופעת המסייע בדמות רוקי. השיגרה newCommandBarAndButton מעבירה הלאה את שני השלבים הראשונים של יצירת סרגלי פקודות אל השיגרה addShowAssistantsAndRocky. הצבת השלבים האלה בשיגרה נפרדת שימושית לאחת הדוגמאות הבאות. השיגרה addShowAssistantsAndRocky מעניקה לסרגל הפקודות החדש את השם Show Assistants. כעת השיגרה מוסיפה פקד מותאם אישית. כאשר מגדירים פקדים לסרגלי פקודות מותאמים אישית, חובה להקצות ערך למאפיין Style לצד ערכי המאפיינים האחרים שמגדירים בסרגלי פקודות מוכללים. אם תשכח לעשות זאת, השיגרה addShowAssistantsAndRocky עלולה להציג בסרגל הפקודות לחצן ריק.

```
Sub newCommandBarAndButton()
On Error GoTo CBarBtnTrap
Dim cbr1 As CommandBar
Dim cbr1btn1 As CommandBarButton
Dim cbr1Name As String
' Add command bar to show Rocky.
   addShowAssistantsAndRocky
' Make CommandBar visible.
   Set cbr1 = CommandBars("Show Assistants")
   cbr1.Visible = True
CBarBtnExit:
   Exit Sub
CBarBtnTrap:
   Debug.Print Err.Number; Err.Description
   Resume CBarBtnExit
End Sub

Sub addShowAssistantsAndRocky()
Dim cbr1 As CommandBar
Dim cbr1btn1 As CommandBarButton
' Add a command bar named Show Assistants.
   Set cbr1 = CommandBars.Add("Show Assistants", msoBarTop, , True)

'Add a button control to the command bar.
   Set cbr1btn1 = cbr1.Controls.Add(msoControlButton, , , , True)
' Set button properties.
   With cbr1btn1
      .Caption = "Show Rocky"
      .BeginGroup = True
      .OnAction = "showRocky"
      .Style = msoButtonCaption
   End With
End Sub
```

לאחר שהשליטה שבה אל השיגרה newCommandBarAndButton, היא מגדירה ערך True למאפיין Visible (גלוי) של הפקד. ללא שלב זה, הדרך היחידה בה יכול משתמש לראות את סרגל הפקודות המותאם אישית החדש היא על ידי הצגתו במפורש (למשל, בלחיצת עכבר ימנית על סרגל פקודות ובחירת שם סרגל הפקודות הרצוי). הקוד ללכידת שגיאות בשיגרה newCommandBarAndButton מאפשר ליישום להפעיל את השיגרה גם כאשר סרגל הפקודות כבר קיים. ללא הלוגיקה ללכידת שגיאות, השיגרה addShowAssistantsAndRocky תחולל שגיאה מכרעת כאשר תנסה ליצור סרגל פקודות שכבר קיים. הואיל ושגיאה זאת אינה קריטית (סרגל הפקודות כבר קיים) ניתן בהחלט להתעלם ממנה.

שינוי סרגלי פקודות מותאמים אישית

שלוש השגרות הבאות מוסיפות פקדים חדשים בסרגל פקודות מותאם אישית קיים. כמו כן הן חושפות גישה נוספת לטיפול בסוגיית סרגל הפקודות הקיים. השיגרה addCbrBtns מציבה זוג לחצנים נוסף בסרגל הפקודות שיצרנו בדוגמה הקודמת. אם סרגל פקודות זה אינו קיים, השיגרה מפעילה את השיגרה addShowAssistansAndRocky. השיגרה addCbrBtns מפעילה באופן מותנה את השיגרה היוצרת את סרגל הפקודות Show Assistants לפי ערך ההחזרה של שגרת הפונקציה doesCbrExist. שיגרה זאת מבררת אם סרגל פקודות נתון כבר קיים. בין אם סרגל הפקודות Show Assistants כבר קיים ובין אם לאו, משפט If...Then...Else ההתחלתי מגדיר הפניה לסרגל זה. יתרת השיגרה מציבה בסרגל הפקודות עוד שני לחצנים. בסיום השיגרה addCbrBtns, סרגל הפקודות הופך נראה לעין אם עד כה היה מוסתר.

```
Sub moreButtons()
   addCbrBtns "Show Assistants"
End Sub

Sub addCbrBtns(cbrName As String)
Dim cbr1 As CommandBar
Dim cbr1btn1 As CommandBarButton

' Optionally create Show Assistants command bar.
' Reference it with a variable.
   If Not doesCbrExist(cbrName) Then
      addShowAssistantsAndRocky
      Set cbr1 = CommandBars(cbrName)
   Else
      Set cbr1 = CommandBars(cbrName)
   End If
' Add a new button to Show Assistants command bar.
   Set cbr1btn1 = cbr1.Controls.Add(msoControlButton, , , , True)
```

```
' Set properties for button to show Clippit.
    With cbr1btn1
        .Caption = "Show Clippit"
        .OnAction = "showClippit"
        .Style = msoButtonCaption
    End With

' Add a new button to Show Assistants command bar.
    Set cbr1btn1 = cbr1.Controls.Add(msoControlButton, , , , True)

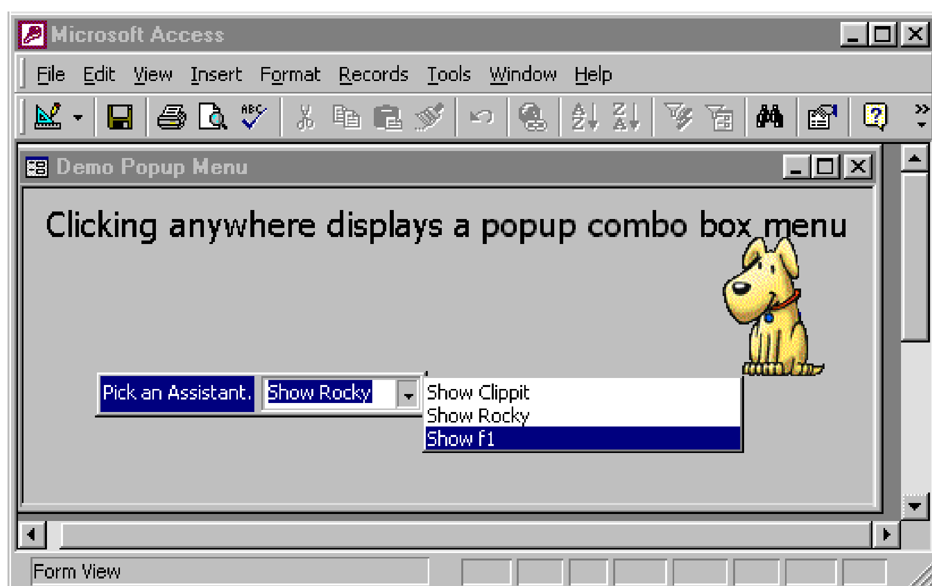
' Set properties for button to show F1.
    With cbr1btn1
        .Caption = "Show F1"
        .OnAction = "showF1"
        .Style = msoButtonCaption
    End With

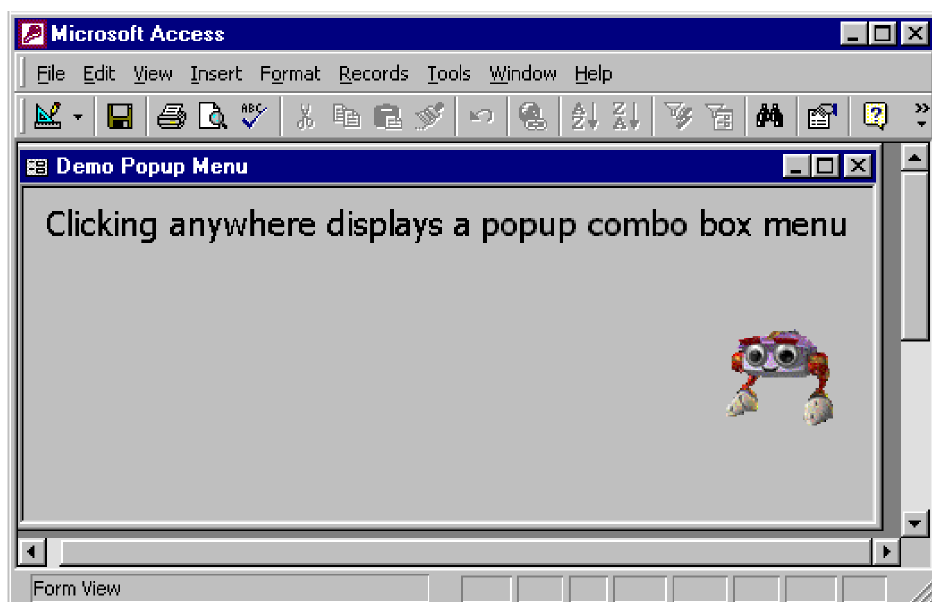
' Make the Show Assistants command bar visible.
    If Not cbr1.Visible = True Then cbr1.Visible = True
End Sub

Function doesCbrExist(cbrName As String) As Boolean
Dim cbr1 As CommandBar
    doesCbrExist = False
    For Each cbr1 In CommandBars
        If cbr1.name = cbrName Then
            doesCbrExist = True
        End If
    Next cbr1
End Function
```

יצירת סרגלי פקודות מוקפצים

דוגמת השיגרה הראשונה שלהלן הופכת לזמין פקד תיבה-משולבת בסרגל פקודות מותאם אישית, והופכת את סרגל הפקודות לשורת תפריטים מוקפצת. תרשים 8.10 שלהלן מציג את התנהגות התפריט המוקפץ על גבי טופס. לחץ בכל נקודה-שהיא בטופס כדי להקפיץ סרגל פקודות מותאם אישית ובו פקד יחיד. פקד זה הוא תיבה משולבת עם ערכים לבחירת מסייע בדמות המהדק (Clippit), רוקי (Rocky) או F1. התהליך מתחיל באירוע לחיצה עבור מקטע הפירוט של הטופס. שלוש השגרות הבאות מכילות את הקוד ליצירת הדוגמה המוצגת בתרשים 8.10.





תרשים 8.10: שורת תפריטים מוקפצת בהתאמה אישית, עם פקד תיבה-משולבת. לפתיחת שורת התפריטים המותאמת אישית, לחץ בכל נקודה-שהיא בטופס

```
Private Sub Detail_Click()
    ShowAndProcessComboBox
End Sub
```

```
Sub showAndProcessComboBox()
Dim cbr1 As CommandBar
' Call from Click Event in form.
    If doesCbrExist("Custom1") Then
        CommandBars("Custom1").ShowPopup
    Else
        createAndShowPopUpMenu
    End If
End Sub

Sub createAndShowPopUpMenu()
Dim cbr1 As CommandBar
' Add command bar named Custom1.
    Set cbr1 = CommandBars _
        .Add(name:="Custom1", Position:=msoBarPopup, Temporary:=True)
    With cbr1
        .Controls.Add Type:=msoControlComboBox
        With .Controls(1)
            .Style = msoComboLabel
            .Caption = "Pick an Assistant."
            .AddItem "Show Clippit"
            .AddItem "Show Rocky"
            .AddItem "Show F1"
            .OnAction = "processComboBoxChoice"
        End With
    End With
    cbr1.ShowPopup
End Sub

Sub processComboBoxChoice()
Dim caseValue As Integer
' Decode selected item and implement corresponding method.
    Select Case CommandBars("custom1").Controls(1).ListIndex
        Case 1
            showClippit
        Case 2
            showRocky
        Case 3
            showF1
    End Select
End Sub
```

השיגרה הראשונה היא שגרת האירוע שמאחורי הטופס, והיא מפעילה את השיגרה showAndProcessComboBox, השוכנת במודול סטנדרטי (standard module). שיגרה זאת קובעת אם סרגל הפקודות Custom1 כבר קיים. אם כן, השיגרה מפעילה את

השיטה ShowPopup כדי להציג את סרגל הפקודות כשורת תפריטים מוקפצת. אם לא, היא יוצרת את סרגל הפקודות Custom1 על ידי הפעלת השיגרה createAndShowPopupMenu. כמשתמע משמה של השיגרה השלישית, היא יוצרת את סרגל הפקודות המותאם אישית ומייד מציגה אותו כשורת תפריטים מוקפצת.

השיגרה createAndShowPopupMenu אמנם קצרה, אולם היא מפעילה טכניקות מעניינות. ראשית, נעשה בה שימוש במשפטי With...End מקוננים. המשפט החיצוני מוסיף חבר חדש לאוסף CommandBars, והמשפט הפנימי מוסיף פקד לאותו חבר. הקצאות המאפיינים בתוך משפט With...End הפנימי מגדירות לפקד סגנון של תיבה משולבת, מגדירות את האלמנטים ברשימת התיבה המשולבת, ומציינות את השיגרה processComboBoxChoice, המופעלת בעקבות בחירה של המשתמש מתוך התיבה המשולבת. שיגרה אחרונה זאת משתמשת במשפט Select Case המבוסס על האלמנט שנבחר מתוך התיבה המשולבת כדי להפעיל אחת משלוש השגרות המותאמות אישית המציגות מסייע.

מחיקת סרגלי פקודות מותאמים אישית

אם אתה עוסק בבניית סרגלי פקודות מותאמים אישית ביישומים, מן הסתם תצטרך לפעמים גם למחוק אחד מהם, או יותר. בדוגמה הבאה נמחק סרגל כלים באמצעות יצירת לולאה העוברת על כל סרגלי הפקודות כדי לאתר את אלה המותאמים אישית, כלומר אלה שבהם המאפיין Builtin מכיל ערך False. כאשר השיגרה מוצאת סרגל פקודות מותאם אישית, היא שואלת את המשתמש אם למחוק אותו. אם המשתמש משיב **כן** (Yes), סרגל הפקודות נמחק והשיגרה מוסיפה אחד למניין סרגלי הפקודות שנמחקו. בכל מקרה, השיגרה מוסיפה אחד למשתנה המונה את סרגלי הכלים המותאמים אישית.

```
Sub deleteCustomCbr()
Dim cbr1 As CommandBar, delFlag As Boolean
Dim delBars As Integer, cusBars As Integer

' Not necessary to initialize delFlag, delBars, or cusBars because their default
' values (False and 0) are OK.

' Conditionally delete custom menu bars.
    For Each cbr1 In CommandBars
        If (cbr1.BuiltIn = False) Then
            If MsgBox("Are you sure that you want to " & _
                "delete the " & cbr1.name & " command bar?", _
                vbYesNo, "Programming Microsoft Access 2000") = _
                    vbYes Then
                    cbr1.Delete
```

```
                delFlag = True
                delBars = delBars + 1
            End If
            cusBars = cusBars + 1
        End If
    Next cbr1

    ' Report outcome of command bar enumeration.
    If Not delFlag Then
        If cusBars > 0 Then
            MsgBox "No custom command bars deleted " & _
                "out of a total of " & cusBars & ".", _
                vbInformation, "Programming Microsoft Access 2000"
        Else
            MsgBox "No custom command bars.", vbInformation, _
                "Programming Microsoft Access 2000"
        End If
    Else
        MsgBox delBars & " custom command bar(s) deleted.", _
            vbInformation, "Programming Microsoft Access 2000"
    End If
End Sub
```

השיגרה deleteCustomCbr מסתיימת בהצגת אחד משלושה משפטים אפשריים, לפי מספר סרגלי הפקודות שנמחקו וסך כל סרגלי הפקודות המותאמים אישית. זוג משפטי If...Then מקוננים מטפל בניתוב אל המשפט המתאים עבור תיבת ההודעה. אם אין מחיקות, אולם יש לפחות סרגל פקודות מותאם אישית אחד, המשפט מציג הודעה המדווחת כי לא נמחקו סרגלי פקודות, ומציגה את מספר סרגלי הפקודות המותאמים אישית. אם אין מחיקות ואין סרגלי פקודות מותאמים אישית, השיגרה מציגה הודעה ברוח זאת. לבסוף, אם השיגרה גילתה סרגלי פקודות, תיבת ההודעה מציגה את מספרם.

9

שילוב Access עם יישומי Office אחרים

בתחילת דרכה, זכתה Microsoft Access לפופולריות עצומה כאחד מרכיבי ערכת התוכנות Microsoft Office. אלמנטים רבים בממשק המשתמש של Access משותפים גם לממשקי המשתמש ביישומי Office אחרים, וקל למדי להעביר נתונים בין Access לשאר מרכיבי Office. נוסף לכך, ניתן לשלב את Access עם שאר מרכיבי Office גם ביישומים מותאמים אישית. דבר זה מאפשר ליהנות מהיתרונות של ערכת מסד-נתונים בסביבה הידידותית והמוכרת של מעבדי תמלילים וגליונות אלקטרוניים.

פרק זה מסביר כיצד לבצע שילוב תוכניתי של Access 2000 עם יישומי Office האחרים בעזרת תכונות Access מוכללות:

- יישומיך יוכלו להגיע למנהלי ISAM באמצעות האובייקט Connection כדי לטפל בנתונים של גליונות אלקטרוניים ב- Microsoft Excel. אובייקט Connection המבוסס על מנהל ISAM יכול לשמש כערוץ דו-כיווני לשיתוף נתונים בין Access ו-Excel.
- תוכל להגיע באופן תוכניתי למקורות נתונים של Access בעזרת יכולת מיזוג הדואר ב- Microsoft Word, המפשטת יצירה של תוויות מען, מכתבים אחידים וקטלוגים של מוצרים.
- בעזרת אוטומציה, יישומיך יוכלו לנצל בו-זמנית את מבני האובייקט משני יישומי Office, או יותר. לדוגמה, יישום יכול לייצא שמות וכתובות מתוך מחסן נתוני Access אל תיקיית **אנשי קשר** (Contacts) ב- Microsoft Outlook. בדומה לכך, ניתן לאכלס ערכים ממקור נתוני Access בטבלאות של מסמך Word.

הדוגמאות בפרק זה מתמקדות ב- Access, ב- Outlook וב- Word, אולם העקרונות הכלליים חלים גם על יישומי Office אחרים, וכן על ערכות של יצרנים אחרים, שנחשפות בהן תבניות האובייקט באמצעות אוטומציה ומתאפשר בהן טיפול בנתונים באמצעות Microsoft Visual Basic for Applications (VBA).

קישור Access ליישומי Office אחרים

בסעיף זה תערוך היכרות עם שלוש טכניקות (מנהלי ISAM ניתנים להתקנה, השיטה OpenDataSource באובייקט MailMerge, ואוטומציה) להפעלת Access בשילוב עם יישומי Office אחרים. בסעיפים הבאים ייעשה שימוש בטכניקות אלה בהקשר מעשי.

מנהלי ISAM ברי התקנה

השתמש באובייקט Connection המוכר ב- ADO (Microsoft ActiveX Data Objects) לקישור עם מקורות נתונים אחרים באמצעות מנהלי ISAM. מקורות מידע כאלה יכולים להיות כאלה שאינם Jet ואינם ODBC, כמו למשל Excel, dBASE ו- Paradox. בסעיף זה נשתמש במנהל ISAM ב- Excel כדי ליצור קישוריות בין Access לחוברות עבודה ב- Excel. יש ליישם טכניקות דומות בכל הנוגע למנהלי ISAM בקבצי dBASE, Paradox, Lotus 1-2-3, טקסט ו- HTML, אלא שלכל מנהל התקן מאפיינים נפרדים ומגבלות משלו. תוכל להרחיב את ידיעותיך על ידי עיון בסיכום אודות המאפיין Connect בתוכנית העזרה המקוונת.



הערה:

היקף התמיכה ל- ISAM בר התקנה ממשיך לגדול בהתאם לצורכי המשתמש וההתפתחויות הטכנולוגיות. ב- Access 2000 בוטלה התמיכה ל- ISAM עבור מסדי נתונים של Microsoft FoxPro לטובת מנהל ODBC FoxPro. מנהלי ISAM המקובלים ימשיכו לפעול עם נתוני dBASE ו- Paradox בגירסה 5 ומטה. אם דרושה לך גישת קריאה/כתיבה לגרסאות אחרות, עליך לרכוש בנפרד את תוכנת Borland Database Engine באמצעות Inprise.

כאשר אתה משתמש במנהל ISAM, מחרוזת החיבור שלך מכילה שלושה ארגומנטים, שכל אחד מהם חייב להסתיים בנקודה-פסיק. תחילה עליך לקבוע ספק. אם אתה משתמש במנהל ISAM בר התקנה, התחל את מחרוזת החיבור בהפניה לספק Jet 4. המשך במיפרט המצביע אל הקובץ עבור מקור הנתונים. אם מדובר ב- Excel, על הפירוט לכלול את הכונן, הנתיב ושם הקובץ. קבע את הפרמטר האחרון על ידי הצבת שם מנהל ISAM הרצוי כהגדרה לפרמטר המאפיינים המורחבים. קיימים מנהלי התקן מוגדרים לגרסאות השונות של Excel ולסוגים אחרים של מקורות נתונים שניתן ליצור עמם קישור. ההפניה אל חוברת עבודה ב- Excel 2000 נעשית באמצעות המחרוזת "Excel 8.0", ולאחריה נקודה-פסיק.

בדוגמת הקוד הפשוטה שלהלן נעשה שימוש במנהל ISAM כדי ליצור קישור עם חוברת עבודה ב- Excel 2000 בתוך יישום של Access 2000. המשפט Dim מצהיר על אובייקט Connection חדש ויוצר אותו. המשפט הבא מתחיל את החיבור על ידי הצבעה אל חוברת עבודה של Excel באמצעות מנהל ISAM ב- Excel 8. לאחר שנוצר החיבור עם מקור הנתונים, על היישום להגדיר טווח של תאים בחוברת העבודה. בדוגמת הקוד

מוקצה הטווח customers שבתוך הקובץ לערכת הרשומות ששמה rst1. Access משתמש בקישור זה כדי לטפל בנתונים שבחוברת העבודה. בסיום הדוגמה מודפסות שתי העמודות הראשונות של השורה הראשונה מתוך הטווח שצוין בחוברת העבודה של Excel בתוך חלון Immediate ב- Access.

```
Sub connect2XLPrintFromFirst()
Dim cnn1 As New ADODB.Connection, rst1 As ADODB.Recordset
' Make connection to Excel source.
   cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=C:\Programming Access\Chap09\customers.xls;" & _
      "Extended Properties=Excel 8.0;"
' Open read-only recordset based on Excel source.
   Set rst1 = New ADODB.Recordset
   rst1.CursorType = adOpenForwardOnly
   rst1.LockType = adLockReadOnly
   rst1.Open "customers", cnn1, , , adCmdTable
' Print selected fields from first record.
   Debug.Print rst1.Fields(0).Value, rst1.Fields(1).Value
' Close connection to source.
   cnn1.Close
End Sub
```

כאשר עובדים עם מנהל ISAM אין צורך בפתיחת מקור הנתונים ב- Excel (או אפילו בפתיחת Excel עצמו). כמו כן לא דרושה ליישום הפניה אל תבנית האובייקט של Excel. על אף הדרישות המזעריות של מנהל ISAM ב- Excel, ניתן להשתמש בו הן לקריאה והן לעדכון של מקורות נתונים ב- Excel.

השיטה OpenDataSource

באפשרותך להשתמש בשיטה OpenDataSource של האובייקט MailMerge כדי ליצור קישור עם מקור נתוני Access מתוך יישום של Word. עליך להשתמש ב- Access – וליתר דיוק, ב- Jet – כמחסן נתונים ליישומי מיזוג דואר היוצרים תוויות מען, מכתבים אחידים, קטלוגים של מוצרים, וכו׳. אם כי ניתן לבצע ב- Access חלק מהפעולות הללו באמצעות האובייקט Report, סביבת Word היא הטבעית יותר לכתיבת תכנים. כמו כן תמצא ב- Word כלים מעולים לעיצוב טקסט ומאפייני WYSIWYG (׳מה שאתה רואה הוא מה שאתה מקבל׳) החסרים באובייקט Report של Access. תוכל לנצל משאבים אלה בעזרת שגרות VBA מבוססות-Word, וכן בעזרת אוטומציה מתוך Access.

כאשר אתה מפנה למקור נתוני Access באמצעות השיטה OpenDataSource, עליך להפנות תחילה לקובץ של מסמך Word, וכן אל האובייקט MailMerge של Word. יש לציין שני פרמטרים עבור השיטה עם Access: הפרמטר Name (שם), המציין את הכונן,

הנתיב ושם הקובץ של מקור הנתונים ב- Access, והפרמטר Connection (חיבור), המגדיר מקור נתונים מסוג 'טבלה' או 'שאילתה', ואת שם אובייקט מסד הנתונים ב- Access. חובה לכלול במסמך Word סימניות או שדות מיזוג דואר המצביעים אל השדות שבמסד נתוני Jet. כדי להפעיל מיזוג השואב נתונים ממקור הנתונים שצוין (למשל, טבלה ב- Access) אל תוך מסמך Word, עליך להפעיל את השיטה Execute עבור האובייקט MailMerge.

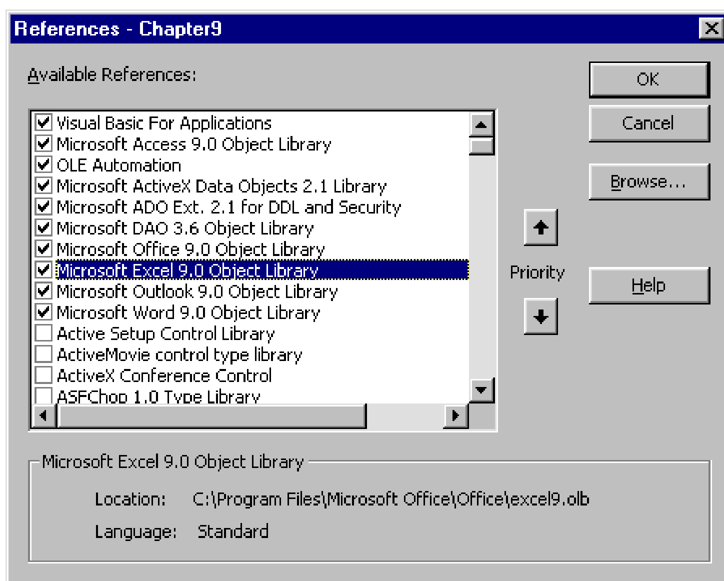
לרשותך מספר דרכים לסינון ערכים המופיעים במסמך מיזוג דואר ב- Word. לדוגמה, תוכל השתמש בפרמטר SQLStatement של השיטה OpenDataSource כדי לציין אילו רשומות לשלוף מתוך מקור הנתונים. כאשר עושים פעולה זאת עם מקור נתוני Jet, ההפניה אל Access נעשית באמצעות מנהל ODBC, תוך הצבת constr כהגדרת הפרמטר Connection. השתמש בתחביר של משפטי SQL לסינון רשומות מתוך טבלה או שאילתה ב- Access.

גישה אחרת לסינון כרוכה בשאילתה מיוחדת במסגרת Access. הפרמטר Connect של השיטה OpenDataSource רק מפנה אל אותה שאילתה. השתמש במאפיינים FirstRecord ו- LastRecord של האובייקט DataSource כדי לציין את הרשומה הראשונה והאחרונה האמורה להופיע בתוך מסמך Word לאחר מיזוג. האובייקט DataSource מצביע אל יעד שצוין על ידי השיטה OpenDataSource.

אוטומציה

בעזרת אוטומציה ניתן להפעיל יישום מסוים כך שישלוט ביישום אחר. Microsoft Component Object Model, או COM, מגדיר את הפרוטוקול עבור יכולת זאת. היישום השולט פועל עם היישום הנשלט על ידי טיפול בשיטות ובמאפיינים החשופים של היישום הנשלט, ותגובה לאירועים המתרחשים בו. כדי לעשות זאת, על היישום השולט להכיל הפניה לספריית האובייקטים של היישום השני, וליצור מופע של אותו יישום (עיין בפרקים 7 ו-8, שם תמצא פרטים על יצירה וניהול של הפניות באמצעות קוד). היישום השולט מפעיל שיטות ומקצה ערכי מאפיינים באמצעות המופע של היישום הנשלט.

תרשים 9.1 מציג תיבת דו-שיח **References** (הפניות) מתוך יישום ב- Access, עם הפניות ל- Excel, Outlook ו- Word, כמו גם לספריית Office המכילה את מבני האובייקטים המשותפים. במובן מסוים, אוטומציה הופכת את כל מבני אובייקט הרכיב של Office למשותפים. Access מסוגל לחשוף את מבנה האובייקט שלו כשרת אוטומציה, ולנצל את מבני האובייקט של יישומים אחרים בכך שהוא פועל כמו לקוח של אוטומציה.



תרשים 9.1: תיבת הדו-שיח **References** ב- Access, ובה הפניות ל- Excel, Outlook ו- Word

הפונקציה CreateObject לעומת הפונקציה GetObject

השתמש בפונקציות CreateObject ו- GetObject ליצירת מופעים של יישומים אחרים. הפונקציה GetObject משמשת לבירור אם כבר יש מופע פתוח של יישום מסוים. אם כן, ניתן ליצור הפניה אליו. אם איש אינו משתמש בפועל במופע, דבר זה יכול להיות קביל. אם יישום שרת האוטומציה עדיין אינו פתוח, או אם אתה מעדיף להימנע משימוש במופע פתוח, השתמש בפונקציה CreateObject כדי ליצור מופע חדש של יישום נתון. תוכל להשתמש בפונקציה GetObject גם כדי לפתוח מופע של יישום, שבתוכו פתוח קובץ מסוים.

שתי השגרות הבאות יוצרות מופע של Excel מתוך יישום ב- Access. השיגרה השנייה, isAppThere, מפעילה קישור מאוחר (קישור בזמן ריצה) כדי לברר אם קיים מופע של יישום Office כלשהו. משתנה objApp עם ציון אובייקט כללי יכול לייצג כל יישום של Office (ואפילו אובייקט COM אחר). השיגרה הראשונה, xlThere משתמשת בקישור מוקדם (קישור בזמן הידור) המשתנה xlApp יכול לייצג רק אובייקט Application של Excel. לא ניתן להחליף את Excel.Application בפונקציה CreateObject או GetObject באובייקט Application אחר של Office, כמו למשל Word.Application. עם זאת, ניתן ליצור שיגרה חדשה ונפרדת, למשל wordThere, שתכלול משתנה שיוצהר כאובייקט מסוג Word.Application. השיגרה החדשה תוכל להפנות אל השיגרה הכללית isAppThere, בדומה לפעולה שעושה השיגרה xlThere.

```
Sub xlThere()
Dim xlApp As Excel.Application
   If isAppThere("Excel.Application") = False Then

' If no, create a new instance.
      Set xlApp = CreateObject("Excel.Application")
      xlApp.Visible = True
   Else

' Otherwise, reference the existing instance.
      Set xlApp = GetObject(, "Excel.Application")
   End If

' If user wants instance closed, close app and set reference to Nothing.
   If MsgBox("Close XL ?", vbYesNo, _
      "Programming Microsoft Access 2000") = vbYes Then
      xlApp.Quit
      Set xlApp = Nothing
   End If
End Sub

Function isAppThere(appName) As Boolean
On Error Resume Next
Dim objApp As Object
isAppThere = True
Set objApp = GetObject(, appName)
   If Err.Number <> 0 Then isAppThere = False
End Function
```

בדרך כלל אוטומציה אינה הופכת יישום של Office לנראה לעין כאשר היא פותחת אותו. אם רצונך שהיישום הנפתח ייראה, יהיה עליך בדרך כלל להגדיר למאפיין Visible (גלוי) של אותו יישום ערך True. יישומים שונים חושפים אובייקטים שונים לאוטומציה. Excel חושף אובייקטים כמו Application, Workbook ו- Worksheet. כמובן, שני האובייקטים האחרונים אינם זמינים ביישומי Office האחרים.

סגירת הפניה של אוטומציה

השיגרה xlThere מסלקת באופן מותנה הפניה ליישום Office אחר. תחילה, יש לסגור את היישום או לצאת ממנו (Excel תומך בשיטה Quit). לאחר מכן יש להגדיר בהפניה ערך Nothing. שני השלבים נחוצים כדי לאחזר את המשאבים שתפסה ההפניה של אוטומציה.

עבודה עם Excel מתוך Access

שתי הדוגמאות שלהלן מראות כמה יכולות ש- ISAM בר התקנה ב- Excel מוסיף ליישומים. הדוגמה השלישית מציגה שיטה פשוטה אך רבת-עוצמה לשימוש באוטומציה. במקום לטפל במישרין באלמנטים מפורטים של מבנה האובייקט של שרת אוטומציה, השיגרה מפעילה שיגרה בתוך שרת האוטומציה. שיגרה זאת, בתורה, מעדכנת את קובץ גיליון העבודה, אולם בזמן ובאופן שנקבע על ידי יישום של Access.

עבודה עם ערכים מתוך גליונות עבודה של Excel

שלוש הדוגמאות הבאות פועלות עם חוברת העבודה המוצגת בתרשים 9.2. שם הקובץ הוא MyGas.xls. ארבע העמודות הראשונות בגיליון1 מכילים נתונים שהוזנו ידנית, וארבע העמודות הבאות מכילות ביטויים המבוססים על ארבע הראשונות. בשורת הנוסחאות מוצג הביטוי עבור ערכים בעמודה MPG. הנתונים שוכנים בטווח ששמו gas. תיבת הדו-שיח **הגדרת שם** (Define Name) מציגה את גבולות הטווח בגיליון1.

Microsoft Excel

File Edit View Insert Format Tools Data Window Help

E2 = =B2/C2

MyGas

	A	B	C	D	E	F	G	H
1	Date	Miles	Gallons	Price per	MPG	cents per	e last refill	Total
2	10/28/1995	281.4	10.811	$ 1.44	26.029	$ 0.055		$ 15.56
3	11/11/1995	260	10.628	$ 1.44	24.464	$ 0.059	14	$ 15.29
4	11/22/1995	230	8.526	$ 1.45	26.976	$ 0.054	11	$ 12.35
5	12/06/1995	298	9.742	$ 1.45	30.589	$ 0.047	14	$ 14.12
6	12/16/1995	274.3	10.099	$ 1.45	27.161	$ 0.053	10	$ 14.63
7	01/07/1996	295.4	10.229	$ 1.45	28.879	$ 0.050	22	$ 14.82
8	01/20/1996	268.29						
9	02/03/1996	247.5						
10	02/17/1996	281.7						
11	02/29/1996	281						
12	03/09/1996	304.29						
13	03/27/1996	286.6						
14	04/28/1996	321.4						
15	05/06/1996	321						
16	05/19/1996	266.1						
17	05/25/1996	361						
18	06/02/1996	290.5						
19	06/15/1996	286.29						
20	06/26/1996	306						
21	07/04/1996	303.39						
22	07/08/1996	376.2						

Define Name

Names in workbook:

gas

gas

OK

Close

Add

Delete

Refers to:

=Sheet1!$A$1:$H$45

Ready

תרשים 9.2: גיליון אלקטרוני של Excel עם טווח נקוב, gas, המשתרע על פני התאים A1 עד H45

הדוגמה הראשונה קוראת ערכים מתוך Excel, מבצעת חישובים שונים ב- Access, ומדפיסה את התוצאות בחלון Immediate. לאחר שנוצר חיבור עם מקור הנתונים של Excel, היישום יכול לטפל במקור הנתונים באמצעות קוד, ממש כפי שהוא מטפל בטבלה פנימית ב- Access. לדוגמה, ניתן למנות את הרשומות בטבלה, או לחשב ערכים על סמך הערכים המופיעים במקור הרשומות. הדוגמה מדפיסה בחלון Immediate את הערכים הקיימים בגיליון העבודה של Excel, לצד התוצאות של ביטויים שחושבו ב- Excel. פעולה זאת מאשרת שחישובים ב- Access יכולים להניב תוצאות זהות לחישובים ב- Excel. יכולת זאת עשויה לצמצם את כמות הנתונים שעל היישום לקרוא מתוך גיליון אלקטרוני עתיר-חישובים.

```
Sub openXLComputePrint()
Dim cnn1 As New ADODB.Connection
Dim rst1 As ADODB.Recordset
Dim computedMPG As Double, computedTotal As Currency

' Make connection to Excel source.
    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Programming Access\Chap09\mygas.xls;" & _
        "Extended Properties=Excel 8.0;"

' Open read-only recordset based on Excel source.
' Recall default is read-only.
    Set rst1 = New ADODB.Recordset
    rst1.Open "gas", cnn1, , , adCmdTable

' Enumerate records and compute with field values.
    Do Until rst1.EOF
        computedMPG = rst1.Fields("Miles") / rst1.Fields("Gallons")
        computedTotal = rst1.Fields("Gallons") * rst1.Fields("Price per Gallon")
        Debug.Print rst1.Fields("Date"), _
            rst1.Fields("Miles"), _
            rst1.Fields("Gallons"), _
            rst1.Fields("Price per Gallon"), _
            rst1.Fields("MPG"), computedMPG, _
            rst1.Fields("days since last refill"), _
            rst1.Fields("Total"), computedTotal
        rst1.MoveNext
    Loop

' Close connection to source.
    cnn1.Close
End Sub
```

השיגרה המשנית מצהירה על אובייקט Connection חדש, יוצרת אותו, ולאחר מכן פותחת אותו. דבר זה הוא בעל חשיבות מכרעת כאשר עובדים עם מנהל ISM, מכיון שזוהי הדרך שבה תנהל את הקישור עם מקור הנתונים שמחוץ ל- Access. מחרוזת החיבור מצביעה אל האובייקט בקובץ MyGas.xls של Excel (קטע מתוך נתוניו מופיע בתרשים 9.2). עליך לסיים את מחרוזת החיבור בציון Extended Properties המצביע אל מנהל ה- ISAM המשמש את היישום שלך. בדוגמה נעשה שימוש במנהל ISAM (Excel 8.0) שפועל עם קבצי חוברת עבודה ב- Excel 2000 וב- Excel 97.

ההפניה Recordset בהמשך השיגרה היא מרכיב קריטי, אם כי סטנדרטי למדי, ביישומי ISAM. על ידי הגדרת ערכת רשומות בחיבור, מתאפשר ליישום לקרוא מתוך, ולכתוב אל תוך, מקור הנתונים המרוחק. אם יישום ה- Access שלך יוצר קישור עם מקור הנתונים של Excel, עליך להשתמש בשיטות AddNew ו- Update של האובייקט Recordset כדי להוסיף שורות חדשות לגיליון אלקטרוני מתוך Access. במיפרט ערכת הרשומות עליך לקבוע איזה חלק של חוברת העבודה יקושר. אם תכלול הפניה לטווח gas, ערכת הרשומות תוכל להשתמש בטווח המוכלל עבור גיליון העבודה. התחביר להפניה אל טבלה חיצונית ב- Excel זהה לתחביר המשמש להפניה לטבלה פנימית ב- Access.



הערה:

אם מוטל על היישום שלך לכתוב אל מקור נתונים ב- Excel מתוך Access, או להכניס בו שינויים, הקפד להגדיר סמן שיתמוך בפונקציונליות זאת (לדוגמה, העבר את הקבוע adOpenKeyset עבור סוג הסמן, ואת הקבוע adLockOptimistic עבור סוג הנעילה). שלא כמו ב- DAO, סמן ברירת המחדל ב- ADO אינו תומך בעדכון.

האלמנט החשוב הבא בשיגרה הוא לולאת Do המונה את כל הרשומות בטווח gas. שתי השורות הראשונות בלולאה מחשבות ביטויים עבור שני ערכים מחושבים. המשתנים computedMPG ו- computerTotal משווים את חישובי Access לאלה של Excel, ומאמתים את יכולתך לטפל בנתונים שנקראו ממקור נתונים ב- Excel. שבע השורות הבאות בלולאת Do מדפיסות בחלון Immediate את ערכי שדות הטבלה מ-Excel, לצד שני משתנים מחושבים עבור כל שורה. הניווט בתוך טבלה ב- Excel זהה לניווט בטבלה פנימית. כמובן, עליך להפעיל שיטת MoveNext בתוך הלולאה כדי להתקדם בין שורות גיליון העבודה.

תרשים 9.3 שלהלן מציג את חמש העמודות הקיצוניות מימין בפלט המופק מן הדוגמה הקודמת. שתי העמודות הראשונות מציגות תוצאות זהות עבור MPG ב- Excel (העמודה הראשונה) וב- Access (העמודה השנייה). דבר זה יפה גם לחשבון הדלק הכולל בכל מילוי. דבר זה מאשר שסוגי הנתונים double ו- currency ב- Access מסוגלים להגיע לתוצאות זהות לאלו של Excel.

Immediate

26.0290444917214	26.0290444917214	Null	15.557	15.557
24.4636808430561	24.4636808430561	14	15.2937	15.2937
26.9763077644851	26.9763077644851	11	12.3542	12.3542
30.5892013960172	30.5892013960172	14	14.1162	14.1162
27.1611050599069	27.1611050599069	10	14.6335	14.6335
28.8786782676703	28.8786782676703	22	14.8218	14.8218
26.3987011709141	26.3987011709141	13	14.6246	14.6246
23.0790749720254	23.0790749720254	14	13.287	13.287
26.3123482159537	26.3123482159537	14	13.0506	13.0506
26.7237280076082	26.7237280076082	12	12.8178	12.8178
30.6682120540214	30.6682120540214	9	12.2934	12.2934
25.8874537078855	25.8874537078855	18	13.6063	13.6063
30.6357830521399	30.6357830521399	32	14.6769	14.6769
30.2174526969783	30.2174526969783	8	15.2865	15.2865
25.7350096711799	25.7350096711799	13	15.1895	15.1895
34.9129593810445	34.9129593810445	6	15.0861	15.0861
32.2347980470484	32.2347980470484	8	13.1485	13.1485
28.8366236905721	28.8366236905721	13	14.3857	14.3857
31.3974964087831	31.3974964087831	11	14.122	14.122
31.8385979641096	31.8385979641096	8	13.7122	13.7122

תרשים 9.3: קטע פלט של השיגרה openXLComputePrint. שים לב לתוצאות החישוב הזהות ב- Access וב- Excel

יצירה דינמית של טבלאות Access המבוססות על גליונות עבודה של Excel

הדוגמה הקודמת חשפה את הערכים בגיליון אלקטרוני באמצעות ערכת רשומות. אם מוטל על היישום שלך לפעול דרך קבע עם נתונים מתוך גיליון אלקטרוני, תוכל לשפר את רמת הביצועים שלו על ידי העתקת הערכים מגיליון העבודה אל תוך טבלה מקומית ב- Access. נוסף לכך, היישום יצמצם בדרך זאת את הביקוש שלו למשאבי חיבור על ידי העתקת הערכים מגליונות עבודה אל תוך טבלאות מקומיות כאשר דרושה לו גישה בו-זמנית למספר טווחים שונים בגליונות עבודה. הדוגמה הבאה יוצרת באמצעות קוד טבלה בעלת אינדקס לטווח של גיליון אלקטרוני, ומאכלסת את הטבלה בערכים מתוך אותו טווח. שים לב שבדוגמה נעשה שימוש בסוג הנתונים החדש Identity כדי לציין את ערכי ההתחלה וערכי הצעד בשדה האינדקס של הטבלה (MyID).

```
Sub createTableFromXL()
On Error GoTo createTableTrap

Dim cnn1 As ADODB.Connection
Dim cnn2 As New ADODB.Connection
Dim rst1 As ADODB.Recordset
Dim rst2 As ADODB.Recordset
Dim cat1 As ADOX.Catalog
Dim tbl1 As ADOX.Table
Dim pk1 As ADOX.Index
```

```
' Set connection, catalog, and table objects.
    set cnn1 = CurrentProject.Connection
    Set cat1 = New ADOX.Catalog
    cat1.ActiveConnection = cnn1
    Set tbl1 = New ADOX.Table
' Define table named gas and append it to the Tables collection.
    With tbl1
        .Name = "gas"
        .Columns.Append "Date", adDate
        .Columns.Append "Miles", adDouble
        .Columns.Append "Gallons", adDouble
        .Columns.Append "PricePerGallon", adCurrency
    End With
    cat1.Tables.Append tbl1
    strSQL = "ALTER TABLE Gas ADD COLUMN MyID Identity(2,2)"
    cnn1.Execute strSQL

    Set pk1 = New ADOX.Index
    With pk1
        .Name = "MyPrimaryKey"
        .PrimaryKey = True
        .Unique = True
        .IndexNulls = adIndexNullsDisallow
    End With
    pk1.Columns.Append "MyID"
    tbl1.Indexes.Append pk1

' Make connection to Excel source.
    cnn2.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
       "Data Source=C:\Programming Access\Chap09\mygas.xls;" & _
       "Extended Properties=Excel 8.0;"

' Open read-only recordset based on Excel source.
' Recall default is read-only.
    Set rst1 = New ADODB.Recordset
    rst1.Open "gas", cnn2, , , adCmdTable

'Open read-write recordset based on local table
'named gas.
    Set rst2 = New ADODB.Recordset
    rst2.ActiveConnection = cnn1
    rst2.CursorType = adOpenKeyset
    rst2.LockType = adLockOptimistic
    rst2.Open "gas"
```

```
    Do Until rst1.EOF
        With rst2
            .AddNew
            .Fields("Date") = rst1.Fields("Date")
            .Fields("Miles") = rst1.Fields("Miles")
            .Fields("Gallons") = rst1.Fields("Gallons")
            .Fields("PricePerGallon") = rst1.Fields("Price Per Gallon")
            .Update
        End With
        rst1.MoveNext
    Loop

createTableExit:
    Exit Sub
createTableTrap:
    If Err.Number = -2147217857 Then
        cat1.Tables.Delete "gas"
        Resume
    Else
        Debug.Print Err.Number; Err.Description
        Resume createTableExit
    End If
End Sub
```

השיגרה המתוארת לעיל ארוכה מכיון שהיא מבצעת מספר פונקציות נפרדות, אם כי קשורות זו בזו. כדי ליצור טבלה מקומית שתכיל ערכי גיליון אלקטרוני של Excel, דרוש לדוגמה זוג אובייקטים: Connection ו- Recordset. אובייקטים אלה מספקים קישוריות בו-זמנית אל גיליון העבודה ואל הטבלה המקומית, כך שהשיגרה מסוגלת להעתיק שורה ממקור נתונים אחד למשנהו. כדי להגדיר באמצעות קוד טבלה מקומית ב- Access, הקוד מצהיר על אובייקטי Catalog, Table ו- Index.

לפני העתקת הנתונים מ- Excel, השיגרה מכינה טבלה מקומית שתקלוט אותם. היא פותחת בהצבת החיבור לפרויקט הנוכחי בהפניה cnn1. הואיל ו-cnn1 מפנה לחיבור הפרויקט המקורי, אין צורך לכלול בהצהרה שלו את מילת המפתח New. מנגד, השיגרה יוצרת מופעים חדשים של האובייקטים Catalog ו- Table (דבר שמשתקף בהצהרות שלהם, בהכללת מילת המפתח New). לאחר מכן השיגרה משתמשת בקוד ADO כדי להגדיר ולצרף שדות שיקלטו את ערכי גיליון העבודה. כדי לציין את ערכי ההתחלה וערכי הצעד לאינדקס, השיגרה חוזרת לקוד SQL. יכולת זאת תלויה לחלוטין בפונקציונליות מוכללת של מנגנון Jet. לפיכך, קוד SQL מותאם ספציפית למנגנון מסד הנתונים Jet. לאחר השלמת הגדרת האינדקס וצירופו לטבלה, השיגרה מכוננת חיבור אל גיליון העבודה (בדוגמה זאת נעשה שימוש באותו גיליון אלקטרוני ששימש בדוגמה הקודמת).

כל ניסיון להגדיר מחדש טבלה קיימת יחולל שגיאה שמספרה 2147217857-. השיגרה מוחקת את הטבלה הישנה וממשיכה בהוספת הטבלה החדשה. ביישום אמיתי, ייתכן שעדיף לשמור את הטבלה הישנה בארכיון.

השיגרה מתכוננת להעתקת הערכים על ידי יצירת שני אובייקטי Recordset – האחד עבור הגיליון האלקטרוני, והשני עבור הטבלה המקומית. הקוד משתמש בסמן ברירת המחדל עבור הגיליון האלקטרוני, מכיון שמוטל עליו רק לקרוא מהגיליון האלקטרוני ערכים לפי הסדר. לעומת זאת נעשה שימוש בסמן מסוג adOpenKeyset עבור הקישור לטבלה המקומית, כך שניתן יהיה להוסיף רשומות. מכיון ש-Access מסוגל לחזור על חישובי Excel עם תוצאות זהות, אין צורך להעתיק שדות מחושבים. דבר זה הופך את ערכי השדה בטבלה לבלתי תלויים זה בזה, כך שהטבלה מנורמלת.

הפעלת שגרות Excel מתוך שגרות Access

בשיגרה הבאה, runXL, Access משתמש בפונקציה GetObject כדי ליצור מופע של האובייקט Application של Excel, המכיל את חוברת העבודה MyGas המוצגת בתרשים *9.2*. השיגרה מגדירה ערך True למאפיין Visible של האובייקטים Application ו-Window. לאחר מכן היא מפעילה את השיטה Run של האובייקט Application עבור השיגרה computeOnGas בתיקיה ThisWorkbook של הקובץ MyGas.xls. אחרי שהשיגרה computeOnGas מקובץ ה- Excel מחזירה את השליטה ל- Access, השיגרה runXL מפעילה את השיטה Save עבור ActiveWorkbook ב- Excel. פעולה זאת שומרת את השינויים ומונעת הופעתה של שאלה אם לבצע שמירה זאת כאשר השורה הבאה מפעילה את השיטה Quit. אם אתה מעוניין לסגור את Excel מבלי לשמור את השינויים ובלא השאלה אם לבצע שמירה, הצב ערך True במאפיין Saved של חוברת העבודה בטרם תפעיל את השיטה Quit (עיין בשורת ההערה כדי לברר מהו התחביר הנכון). איחזור משאבי האוטומציה נעשה על ידי הצבת ערך Nothing בהפניה של אובייקט האוטומציה.

```
Sub runXL()
Dim myXLWrkBk As Excel.Workbook
' Open connection to XL workbook and make visible.
    Set myXLWrkBk = GetObject("c:\Programming Access\Chap09\MyGas.xls")
    myXLWrkBk.Application.Visible = True
    myXLWrkBk.Application.Windows("MyGas.xls").Visible = True
' Run procedure in ThisWorkBook folder.
    myXLWrkBk.Application.Run "ThisWorkBook.computeOnGas"
' Close automation object.
' Either invoke the Save method or set the Saved
' property to True to avoid a prompt about saving changes.
    myXLWrkBk.Application.ActiveWorkbook.Save
'     myXLWrkBk.Application.ActiveWorkbook.Saved = True
    myXLWrkBk.Application.Quit
    Set myXLWrkBk = Nothing
End Sub
```

תרשים 9.4 מציג את גיליון העבודה לאחר הפעלת השיגרה computeOnGas. שים לב שהשיגרה מחשבת נתוני סיכום שתי שורות מתחת לשורה האחרונה בטבלה, ומוסיפה עמודה חדשה המציגה את הקילומטראז' היומי (במיילים) בין מילוי מיכל למילוי הבא. השיגרה מתאימה את גודל העמודות לפי הערך הרחב ביותר שלהן.

Microsoft Excel - MyGas

File Edit View Insert Format Tools Data Window Help

I47 = 30.097

	A	B	C	D	E	F	G	H	I
1	Date	Miles	Gallons	Price per Gallon	**MPG**	cents per mile	days since last refill	Total	Miles per Day
32	10/11/1996	346.1	10.407	$ 1.38	**33.256**	$ 0.041	8	$ 14.35	43.26
33	10/20/1996	323.1	10.140	$ 1.38	**31.864**	$ 0.043	9	$ 13.98	35.9
34	10/30/1996	291.79	9.365	$ 1.42	**31.158**	$ 0.046	10	$ 13.29	29.18
35	11/05/1996	291.6	9.606	$ 1.36	**30.356**	$ 0.045	6	$ 13.05	48.6
36	11/11/1996	313.7	9.695	$ 1.36	**32.357**	$ 0.042	6	$ 13.18	52.28
37	11/18/1996	299.39	9.705	$ 1.36	**30.849**	$ 0.044	7	$ 13.19	42.77
38	11/27/1996	336.3	10.721	$ 1.46	**31.368**	$ 0.047	9	$ 15.64	37.37
39	12/05/1996	286.5	9.510	$ 1.32	**30.126**	$ 0.044	8	$ 12.54	35.81
40	12/12/1996	324	10.554	$ 1.30	**30.699**	$ 0.042	7	$ 13.71	46.29
41	12/17/1996	330.1	10.740	$ 1.34	**30.736**	$ 0.044	5	$ 14.38	66.02
42	12/22/1996	318	9.717	$ 1.28	**32.726**	$ 0.039	5	$ 12.43	63.6
43	01/07/1997	361	10.993	$ 1.30	**32.839**	$ 0.040	16	$ 14.28	22.56
44	01/12/1997	347	10.314	$ 1.32	**33.644**	$ 0.039	5	$ 13.60	69.4
45	01/18/1997	313	10.217	$ 1.33	**30.635**	$ 0.043	6	$ 13.58	52.17
46									
47	Summary	13483.5	443.828	1.380191903	**30.38**	0.045	448	$612.57	30.097

Sheet1 / Sheet2 / Sheet3

Ready

תרשים 9.4: קטע פלט של השיגרה computeOnGas; שים לב לעמודת הנתונים החדשה, ולשינוי ברוחב העמודות

השיגרה computeOnGas כרוכה בקוד VBA סטנדרטי בלבד, אולם היא משתמשת באובייקטים, במאפיינים ובשיטות ייחודיים ל- Excel. אם תבצע פעולת אוטומציה, תידרש לך באורח בלתי נמנע היכרות כלשהי עם מבנה אובייקט נוסף, אחד לכל הפחות, במקרה זה מבנה האובייקט עבור יישום Office שאתה מבצע בו פעולת אוטומציה. אחד היתרונות של השימוש בשיטה Run, כמו בשיגרה runXL, הוא שהשיטה מאפשרת למפתחים להתמחות במבני אובייקט מוגדרים. כאשר מפתח מעוניין להשתמש בפונקציה רגילה ביישום בלתי מוכר, באפשרותו להעתיק שיגרה שעוצבה בידי מפתח אחר. גם בלא היכרות יסודית עם יישום מסוים, המפתח יוכל להפעיל את השיטה Run עבור השיגרה שהעתיק.

```
Sub computeOnGas()
Dim mySheet As Worksheet
Dim iRow As Integer, lastRow As Integer
Dim sumDays As Long

' Set reference to first worksheet.
   Set mySheet = Worksheets(1)
   lastRow = Range("gas").Rows.Count
```

```
' Assign column heading.
   mySheet.Cells(1, 9) = "Miles per Day"
   ' Compute miles per day.
   For iRow = 3 To lastRow
      mySheet.Cells(iRow, 9) = _
        Format(Range("gas").Cells(iRow, 2) / Range("gas").Cells(iRow, 7), "0.##")
      sumDays = sumDays + mySheet.Cells(iRow, 7)
   Next iRow
' Compute summary statistics.
   mySheet.Cells(Range("gas").Rows.Count + 2, 1).Select
   ActiveCell.Formula = "Summary"
' Compute total miles.
   ActiveCell.Offset(0, 1).Activate
   ActiveCell.Formula = "=Sum(b2:b" & lastRow & ")" & ""
' Compute total gallons.
   ActiveCell.Offset(0, 1).Activate
   ActiveCell.Formula = "=Sum(c2:c" & lastRow & ")" & ""
' Compute total gas dollars.
   ActiveCell.Offset(0, 5).Activate
   ActiveCell.Formula = "=Sum(h2:h" & lastRow & ")" & ""
' Compute days since last refill.
   ActiveCell.Offset(0, -1).Activate
   ActiveCell.Formula = "=Sum(g3:g" & lastRow & ")" & ""
' Compute price per gallon.
   mySheet.Cells(Range("gas").Rows.Count + 2, 4).Select
   ActiveCell.Formula = "=H" & (lastRow + 2) & "/C" & (lastRow + 2)
' Compute miles per gallon.
   ActiveCell.Offset(0, 1).Activate
   ActiveCell = Format(mySheet.Cells(lastRow + 2, 2) / _
       mySheet.Cells(lastRow + 2, 3), "0.###")
   ActiveCell.Font.Bold = True
' Compute cents per mile.
   ActiveCell.Offset(0, 1).Activate
   ActiveCell = Format(mySheet.Cells(lastRow + 2, 8) / _
       mySheet.Cells(lastRow + 2, 2), "0.###")
' Compute miles per day.
   ActiveCell.Offset(0, 3).Activate
   temp = mySheet.Cells(lastRow + 2, 2)
   temp2 = sumDays
   ActiveCell = Format(mySheet.Cells(lastRow + 2, 2) sumDays, "0.###")
' Resize columns to show values.
  Worksheets("Sheet1").Columns("a:I").AutoFit
End Sub
```

עבודה עם Outlook מתוך Access

תוכנת Outlook משווקת עם ערכת תיקיות סטנדרטית, הכוללת תיקיות ללוח שנה, אנשי קשר, פריטים שנמחקו, טיוטות, תיבת דואר אלקטרוני נכנס, יומן, הערות, תיבת דואר אלקטרוני יוצא, פריטי דואר שנשלחו ומשימות. המשתמש יכול להוסיף תיקיות מותאמות אישית, ולקנן תיקיות זו בזו. המשתמש מבצע פעולות בפריטים בתוך התיקיות – הוספה, מחיקה, תצוגה ופונקציות נוספות.

הגירסה הראשונית של Outlook שווקה עם תמיכה תוכניתית רק באמצעות script של Microsoft VB. ב- Outlook 2000 מתווספת שליטה תוכניתית, בעזרת VBA. Outlook 2000 תומך ביצירת script הן באמצעות VBA והן באמצעות VB Script. למען תאימות עם יתרת הספר, הסעיף מתמקד ביצירת script ב- Outlook מתוך Access, בעזרת VBA. נוסף לכך, בכל הדוגמאות נעשה שימוש בתיקיה **אנשי קשר** (Contacts) כדי ליצור הקשר מוכר.

באפשרותך ליצור ב- Access הפניה למופע של Outlook באמצעות הפונקציה CreateObject. בטרם תוכל להפנות לתיקיה מסוימת, בדרך כלל עליך להחיל את השיטה GetNameSpace על האובייקט Application. האובייקט NameSpace הוא אובייקט בסיס מופשט שמתקיים בין האובייקט Application לבין התיקיות הנפרדות. השיטה מקבלת ארגומנט יחיד, שעליו להיות MAPI במהדורה הנוכחית. עליך להחיל את השיטה GetDefaultFolder על האובייקט NameSpace כדי לקבל את תיקיית ברירת המחדל של סוג נתון. השתמש בקבוע כדי להקצות תיקיית ברירת מחדל שהיישום שלך יטפל בה. הקבוע לתיקיה **אנשי קשר** הוא olFolderContacts.

ספירת פריטים בתיקיה אנשי קשר

השיגרה הבאה מטפלת בתיקיה **אנשי קשר** וסופרת את הפריטים שהיא מכילה. הכן תיקיית **אנשי קשר** לדוגמה, וכלול בה מספר ערכים כדי לבדוק את אופן פעולתה של דוגמה זאת והדוגמאות הבאות. התקליטור המצורף לספר זה כולל נתוני 'אנשי קשר' לדוגמה, לצורך איכלוס של תיקיית **אנשי קשר**.

```
Sub listContacts()
Dim myOlApp As Outlook.Application
Dim myNameSpace As NameSpace
Dim myContacts As Items
Dim myItem As ContactItem
' Create an instance of Outlook.
' Reference its MAPI Namespace.
' Reference MAPI's Contact folder.
    Set myOlApp = CreateObject("Outlook.Application")
    Set myNameSpace = myOlApp.GetNamespace("MAPI")
    Set myContacts = _
        myNameSpace.GetDefaultFolder(olFolderContacts).Items
```

```
' Enumerate items in Contact folder and print selected fields.
   For Each myItem In myContacts
      Debug.Print myItem.FirstName, myItem.LastName, myItem.Email1Address
   Next
End Sub
```

השיגרה פותחת בהצהרה על ארבעה משתנים: אחד ליישום Outlook, אחד לאובייקט NameSpace שלו, אחד לאוסף הפריטים שבתיקיה **אנשי קשר**, ואחד לספירת אותם פריטים. דרושים שלושה משפטי Set כדי לחשוף את פריטי התיקיה **אנשי קשר**. האחרון משתמש בשיטה GetDefaultFolder כדי לחזור לתיקיה **אנשי קשר**, וכן במאפיין Items כדי לגשת אל הפריטים הנפרדים. הספירה מתרחשת בלולאת For...Each. לפריטים בתיקיה **אנשי קשר** סדרה של מאפיינים ובהם פרטים אודות אנשי קשר. בדוגמה נעשה שימוש בשלושה מהמאפיינים האלו כדי להדפיס שם פרטי, שם משפחה וכתובת דואר אלקטרוני ראשונה עבור כל ערך בתיקיה **אנשי קשר**.

הוספת פריט לתיקיה אנשי קשר

באפשרותך לבנות גם פתרונות המבוססים על Access, המטפלים בתכולת התיקיה **אנשי קשר**. מבין שלוש השגרות הבאות, הראשונה, addOneContact, מוסיפה איש קשר חדש בתיקיה. השיגרה משתמשת בקבועי מחרוזת להגדרת השם הפרטי, שם המשפחה וכתובת דואר אלקטרוני עבור איש קשר, אולם אין קושי לשנות את השיגרה כך שתעביר נתונים אלה כארגומנטים. זה בדיוק מה שעושות שתי השגרות הבאות, removeOneEmail ו- deleteAContact. השיגרה removeOneEmail מעבירה כתובת דואר אלקטרוני לשיגרה deleteAContact, מאתרת פריט 'איש קשר' בעל כתובת תואמת, ומוחקת פריט זה.

```
Sub addOneContact()
Dim myOlApp As Outlook.Application
Dim myItem As ContactItem
' Create an instance of Outlook.
   Set myOlApp = CreateObject("Outlook.Application")
' Create an item for the folder.
' Populate the item with values.
' Save the item.
   Set myItem = myOlApp.CreateItem(olContactItem)
   With myItem
      .FirstName = "foo"
      .LastName = "bar"
      .Email1Address = "foobar@yourcompany.com"
      .Save
   End With
End Sub
```

```
Sub removeOneEmail()
   deleteAContact ("foobar@yourcompany.com")
End Sub

Sub deleteAContact(strEmail)
Dim myOlApp As Outlook.Application
Dim myNameSpace As NameSpace
Dim myContacts As Items
Dim myItem As ContactItem
' Create an instance of Outlook.
' Reference its MAPI Namespace.
' Reference MAPI's Contact folder.
   Set myOlApp = CreateObject("Outlook.Application")
   Set myNameSpace = myOlApp.GetNamespace("MAPI")
   Set myContacts = _
      myNameSpace.GetDefaultFolder(olFolderContacts).Items
' Enumerate to search for item to delete.
   For Each myItem In myContacts
      If myItem.Email1Address = strEmail Then
         myItem.Delete
         Exit Sub
      End If
   Next
' No entry found
   MsgBox "No entry found with email of " & strEmail, vbCritical, _
      "Programming Microsoft Access 2000"
End Sub
```

לשיגרה זאת דרושים שני אובייקטים בלבד – אובייקט Application של Outlook, ואובייקט ContactItem שייצג פריט בתיקיה **אנשי קשר**. השיגרה יוצרת הפניה לאובייקט Application בעזרת הפונקציה CreateObject. הפניה זאת תומכת בשיטה CreateItem, שיוצרת מופע ריק של פריט עבור כל תיקיה שצוינה. עליך לקבוע את סוג התיקיה של הפריט באמצעות קבוע שיועבר לשיטה CreateItem. להגדרת מאפייניו של איש הקשר, בחר מתוך רשימה של מעל 140 מאפיינים. הדוגמה מקצה קבועי מחרוזת עבור המאפיינים FirstName, LastName ו- Email1Address (בהחלט ייתכן שלאיש קשר מסוים יותר מכתובת דואר אלקטרוני אחת). לאחר מכן השיגרה מפעילה את השיטה Save כדי לשמור את הערך החדש בתיקיה **אנשי קשר**.

מחיקת פריט מהתיקיה אנשי קשר

השיגרה deleteAContact (המופיעה לעיל) מקבלת כארגומנט מחרוזת המייצגת את ערך המאפיין Email1Address של פריט 'איש קשר' שיש למוחקו. השיגרה מונה את איברי התיקיה **אנשי קשר**, עד שהיא מאתרת פריט בעל מאפיין Email1Address תואם

לארגומנט המועבר. כאשר נמצאת התאמה, השיגרה מוחקת את הפריט על ידי הפעלת השיטה Delete, ויוצאת מהשיגרה כדי למנוע את המשך החיפוש. אם נמנו כל פריטי התיקיה **אנשי קשר** מבלי שנמצא ערך תואם, השליטה עוברת למשפט של תיבת הודעה, המדווחת שלא נמצאו ערכים תואמים לכתובת הדואר האלקטרוני שהועברה לשיגרה.

הוספת פריטים מרובים לתיקיה אנשי קשר

אחת המשימות הנפוצות המבוצעות בעזרת מנהל מסד נתונים, כמו למשל Access, היא הוספת פריטים מרובים לתיקיה **אנשי קשר**. פריטים אלה יכולים להתווסף ממקורות מגוונים, כמו למשל התיקיה **אנשי קשר** במחשב אחר, כתובות שהוזנו דרך האינטרנט, ואפילו קובץ אנשי קשר ישן ב- Access. השיגרה addContacts המובאת להלן משתמשת באחת הגישות לעדכון תיקיית **אנשי קשר** של Outlook בנתוני אנשי קשר מתוך טבלה של Access.

```
Sub addContacts()
Dim myOlApp As Outlook.Application
Dim myItem As ContactItem
Dim rst1 As New Recordset
' Open the Contacts folder in Outlook.
    Set myOlApp = CreateObject("Outlook.Application")

' Open the table with the new contacts.
    With rst1
        .ActiveConnection = CurrentProject.Connection
        .Open "oe4pab"
    End With
' Create a ContactItem for adding contacts and
' loop through the table records to add them to the folder.
    AssistantWorkingOn
    Do Until rst1.EOF
        Set myItem = myOlApp.CreateItem(olContactItem)
        With myItem
            .FirstName = IIf(IsNull(rst1.Fields(0)), "", rst1.Fields(0))
            .LastName = rst1.Fields(1)
            .Email1Address = rst1.Fields(2)
            .Save
        End With
        rst1.MoveNext
    Loop
    AssistantIdleOn
End Sub
```

השיגרה מציבה הפניה ליישום של Outlook, ולאחר מכן פותחת ערכת רשומות המבוססת על הטבלה oe4pab. זוהי הטבלה המקומית בתיקיה **טבלאות** (Tables) שב-Access. הטבלה מכילה 34 ערכים בלבד, אולם השיגרה מסוגלת לקלוט רשימה ארוכה הרבה יותר של כתובות. מסיבה זאת השיגרה מפעילה שיגרה נוספת, המפעילה את המסייע בהנפשת 'עבודה' ומשאירה אותו במצב פעיל עד ש- Access ו- Outlook יסיימו את עדכון התיקיה **אנשי קשר** ב- Outlook בערכים מתוך הטבלה oe4pab (פרק 8 מתאר כיצד לבנות שגרות שישלטו בהנפשת המסייע. תוכל למצוא את השגרות הללו בספריה Chap08 בתקליטור הנלווה). בין שתי הקריאות, האחת להפעלת הנפשת המסייע והשנייה לביטולה, לולאת Do עוברת במחזוריות על כל הרשומות שבטבלה oe4pab. הלולאה יוצרת אובייקט ContactItem חדש בכל מעבר, ולאחר מכן היא מציבה את הרשומות ממעבר זה בפריט, ושומרת את הפריט.

מחיקת פריטים מרובים מהתיקיה אנשי קשר

השיגרה הבאה, removeEmails, היא גירסה מותאמת של השיגרה deleteAContact שהובאה מוקדם יותר. שיגרה זאת מוחקת רשומות מרובות מתוך תיקיית **אנשי קשר**. היא מוחקת את הפריטים אחד לאחד, על ידי קריאות חוזרות לשיגרה deleteAContact, כל פעם עם כתובת אלקטרונית אחרת. הדוגמה משתמשת בכתובות שבטבלה oe4pab כמקור לארגומנטים. לשיגרה זאת שני יתרונות: קל לכתוב אותה, ויש בה ניצול חוזר של השיגרה deleteAContact.

```
Sub removeEmails()
Dim rst1 As New Recordset
' Open the table with the new contacts.
    With rst1
        .ActiveConnection = CurrentProject.Connection
        .Open "oe4pab"
    End With
    ' Loop through the table records to add them to the folder.
    AssistantWorkingOn
    Do Until rst1.EOF
        deleteAContact (rst1.Fields(2))
        rst1.MoveNext
    Loop
    AssistantIdleOn
End Sub
```

אם כי שיגרה זאת יכולה לבצע את המשימה, יש בה לפחות שני חסרונות. ראשית, השיגרה עוברת על כל ערכי התיקיה **אנשי קשר** עבור כל פריט המיועד למחיקה. דבר זה הופך יקר יותר בהתמדה ככל שגדל מספר הפריטים המיועדים למחיקה, מספר הפריטים בתיקיה עצמה, או שניהם גם יחד. שנית, אם לא נמצא פריט תואם, השיגרה מציגה תיבת הודעה המחייבת את המשתמש ללחוץ על לחצן כדי להמשיך. אם מספר

רב של ערכים ברשימת הפריטים למחיקה אינו נמצא עוד בתיקיה **אנשי קשר**, הכורח ללחוץ על לחצן עבור כל פריט שחסר בתיקיה יכול להיות מייגע. אחת הדרכים להתגבר על חסרונות אלה היא להחליף את השיגרה deleteAContact בקריאה לשיגרה deleteAContact2, המוצגת להלן.

```
Sub deleteAContact2(strEmail)
On Error GoTo delete2Trap
Dim myOlApp As Outlook.Application
Dim myNameSpace As NameSpace
Dim myContacts As Items
Dim myItem As ContactItem
Dim strFilter As String
' Create an instance of Outlook.
' Reference its MAPI Namespace.
' Reference MAPI's Contact folder.
    Set myOlApp = CreateObject("Outlook.Application")
    Set myNameSpace = myOlApp.GetNamespace("MAPI")
    Set myContacts = myNameSpace.GetDefaultFolder(olFolderContacts).Items

' Find target item and remove it.
    strFilter = "[Email1Address] = """ & strEmail & """"
    Set myItem = myContacts.Find(strFilter)
    myItem.Delete
delete2Exit:
    Exit Sub
delete2Trap:
    If Err.Number = 91 Then
' If item is not there, just keep on going.
        Resume Next
    Else
' Otherwise, pause with a message box.
        MsgBox Err.Number & ": " & vblfcr & Err.Description, vbCritical, _
            "Programming Microsoft Access 2000"
        Resume Next
    End If
End Sub
```

שיגרה זאת מחישה את החיפוש אחר פריט המיועד למחיקה באמצעות השיטה Find. היא בונה קריטריון לשיטה Find, המבוסס על הכתובות האלקטרוניות המועברות אליה. לאחר שהיא מאתרת פריט בעל כתובת אלקטרונית תואמת, השיגרה מחילה את השיטה Delete על אותו פריט. השימוש ב- Find מניב חיסכון ניכר בזמן, גם כאשר מדובר ברשימת פריטים קצרה, כמו זאת שבטבלה oe4pab, אולם יתרון המהירות גדל ככל שגדלה רשימת הכתובות האלקטרוניות, או מספר הפריטים, כמו בתיקיה

אנשי קשר. השיגרה לוכדת גם מקרי כשל, כאשר השיטה Find אינה מחזירה פריט. דבר זה קורה כאשר אין בתיקיה **אנשי קשר** פריט תואם לפי כתובת אלקטרונית. במצב כזה, השיגרה deleteAContact2 מחזירה בשקט את השליטה לשיגרה הקוראת, כדי שזו תוכל לחפש כתובת אלקטרונית חדשה, ללא צורך בהתערבות המפעיל.

עבודה עם Word מתוך Access

הדוגמה הראשונה בסעיף זה מראה כיצד לבנות ולעצב טבלה ב- Word, ולאכלס אותה בנתונים מתוך טבלה ב- Access. בדוגמה נעשה שימוש באוטומציה כדי לשלוט ב- Word מתוך Access, והיא כוללת אפילו טופס Access פשוט להפעלת שגרת האוטומציה. שתי הדוגמאות הבאות מטפלות באמצעות קוד בשתי משימות של מיזוג דואר: הפקת תוויות מען, והפקת מכתב אחיד.

דוגמת המכתב האחיד המובאת כאן מבצעת פעולה זהה לזו של דוגמת המכתב האחיד בפרק 6. תוכל להשוות בין שתי הגישות כדי לברר איזו מתאימה יותר לצרכיך. ככלל, הרעיון שמאחורי רכיבים מרובים הוא שרצוי להשתמש בכל אחד מהם לביצוע המשימה שהוא מיטיב לבצע. הדוגמה בפרק זה מאפשרת שמירת נתונים ב- Access, והפקת מכתבים אחידים להדפסה ב- Word. אם אתה מעדיף לעבוד עם מבנה האובייקט של Word ועם האובייקט MailMerge, ייתכן שגישה זאת תיטיב לשרת אותך. לעומת זאת, אם אתה מעדיף להתרכז בהתמחות ב- Access, ייתכן שעדיפה הגישה המובאת בפרק 6.

אוטומציה של Word מתוך Access

דוגמת האוטומציה הבאה מעבירה תכנים מערכת רשומות המבוססת על טבלה ב- Access, לטבלה ב- Word. כאשר יש לך הפניות למבני אובייקטים מרובים בעלי מונחים דומים, עליך להוסיף בהצהרה על האובייקט קידומת לפני הקצאת סוג הנתונים של האובייקט: לדוגמה, השתמש ב- Word.Table במקום ב- Table. פעולה זאת מורה למתרגם VBA איזה אובייקט Table דרוש לך. זכור, שגם Access יכול להצהיר על אובייקט Table מתוך ספריית ADOX. זכור גם שהתנהגות האובייקט Range ב- Word שונה מהתנהגותו ב- Excel. VBA ו- IntelliSense מאפשרים לך לבנות פתרונות על פני מספר יישומים, אולם אין הם פוטרים אותך מהכרת מבני האובייקט ביישומים השונים.

```
Sub fromAccessToWordTable()
Dim myWDApp As Word.Application
Dim myRange As Word.Range, myTable As Word.Table
Dim acell As Word.Cell, emailCol As Integer
Dim rst1 As New Recordset, irow As Integer
```

```
' Open the table with the new contacts.
    With rst1
        .ActiveConnection = CurrentProject.Connection
        .Open "oe4pab", , adOpenKeyset, adLockOptimistic, adCmdTable
    End With

' Create a Word application instance and turn on
'the Assistant's working animation.
    AssistantWorkingOn
    Set myWDApp = CreateObject("Word.Application")

' Add a document to the application and a table to the document.
' Specify rows to equal one more than e-mail address table in Access.
    myWDApp.Documents.Add
    Set myRange = myWDApp.ActiveDocument.Range(0, 0)
    myWDApp.ActiveDocument.Tables.Add Range:=myRange, _
        NumRows:=rst1.RecordCount + 1, NumColumns:=3

' Insert column headings for table.
    With myWDApp.ActiveDocument.Tables(1).Rows(1)
        .Cells(1).Range.Text = rst1.Fields(0).Name
        .Cells(2).Range.Text = rst1.Fields(1).Name
        .Cells(3).Range.Text = rst1.Fields(2).Name
    End With

' Insert first name, last name, and e-mail from Access table.
' Insert contact info in the second through the last row.
    For irow = 2 To myWDApp.ActiveDocument.Tables(1).Rows.Count
    emailCol = 0
        For Each acell In myWDApp.ActiveDocument.Tables(1).Rows(irow).Cells
          acell.Range.Text = IIf(IsNull(rst1.Fields(emailCol)), "", _
          rst1.Fields(emailCol))
          emailCol = emailCol + 1
        Next acell
    rst1.MoveNext
    Next irow

' Format table to fit content, turn on idle animation, and
' make Word visible so user can see table in Word.
    myWDApp.ActiveDocument.Tables(1).AutoFitBehavior wdAutoFitContent
    AssistantIdleOn
    myWDApp.Visible = True

End Sub
```

השיגרה מתחילה בפתיחת אובייקט ערכת רשומות המבוסס על טבלה ב- Access, הטבלה oe4pab המוכרת מהדוגמאות על עבודה עם Outlook מתוך Access. מכיון שהיישום משתמש במאפיין RecordCount, עליך להימנע משימוש בסמן מסוג קדימה-בלבד (forward-only). בדוגמה נעשה שימוש בקבוע adOpenKeyset לציון סוג הסמן. לאחר פתיחת ערכת הרשומות, היישום מפעיל את הנפשת העבודה של המסייע, ומריץ את הפונקציה CreateObject כדי ליצור מופע חדש של Word.

בשלב הבא, השיגרה בונה את הטבלה ב- Word. היא יוצרת מסמך חדש ומוסיפה טבלה בפינה הימנית העליונה שלו (בממשק עברי). הפרמטרים של השיטה Add באובייקט Table קובעים שבטבלה החדשה מספר השורות יהיה גדול באחד ממספר השורות בטבלה של Access. דבר זה מותיר מקום (שורה אחת) לכותרות העמודות, נוסף לכל הנתונים בטבלה oe4pab. לפני שהיא מתחילה לטפל בערכי ערכת הרשומות, השיגרה כותבת את כותרות העמודות בשורה הראשונה. כותרות אלה הן שמות השדות בטבלה של Access.

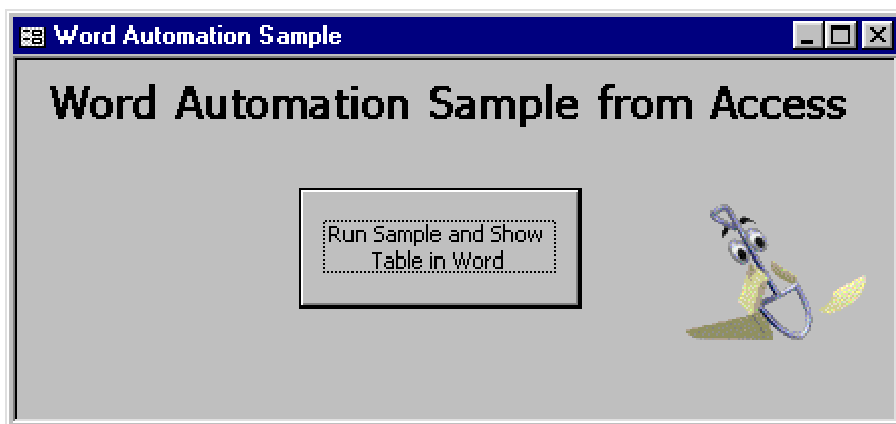
זוג לולאות For מקוננות מסיירות על פני כל התאים בטבלה. הלולאה החיצונית מתקדמת בין השורות, לפי הסדר. הלולאה הפנימית 'צועדת' לרוחב העמודות בתוך שורה נתונה. שים לב, שב- Word קיים אוסף Cells עבור העמודות בתוך שורה נתונה בטבלה. הלולאה הפנימית מסיירת בין התאים הנפרדים בתוך הטבלה של Word. ההפניה אל תא נפתחת באובייקט האוטומציה myWDApp, ולאחר מכן עוברת באופן הירכי לאובייקט ActiveDocument, לטבלה הראשונה באותו מסמך, ולשורה נתונה בתוך הטבלה. לאחר שזוהה תא המיועד לטיפול בתוך שורה מסוימת, פונקציית IIf המתבססת על ערכי ערכת הרשומות מחשבת ערך עבור המאפיין Text של אותו תא.

לאחר שהשיגרה עוברת במחזוריות על פני כל תאי הטבלה, היא מסתיימת בכך שהיא מבצעת שלוש פעולות. תחילה היא מעצבת מחדש את רוחב העמודות כך שיהיו רחבות די הצורך להצגת ערכי העמודות ללא גלישת שורות. שנית, היא מקצה הנפשת מצב לא-פעיל למסייע. בפועל, דבר זה מפסיק את הנפשת העבודה המתחילה ממש לפני הפעלת האובייקט Application של Word. לבסוף, השיגרה מציבה ערך True במאפיין Visible של האובייקט Application. פעולה זאת מעבירה את המיקוד מ- Access ל- Word.

תרשים 9.5 מציג טופס של Access המאפשר למשתמש להפעיל את השיגרה בלחיצה על לחצן. התרשים מראה את הטופס מייד לאחר שהמשתמש לחץ על הלחצן. דמות המסייע עדיין נמצאת בהנפשת העבודה שלה. כאשר המסייע עובר למצב לא-פעיל, המיקוד עובר מהטופס של Access למסמך Word המכיל את הטבלה שיצרה השיגרה fromAccessToWordTable.

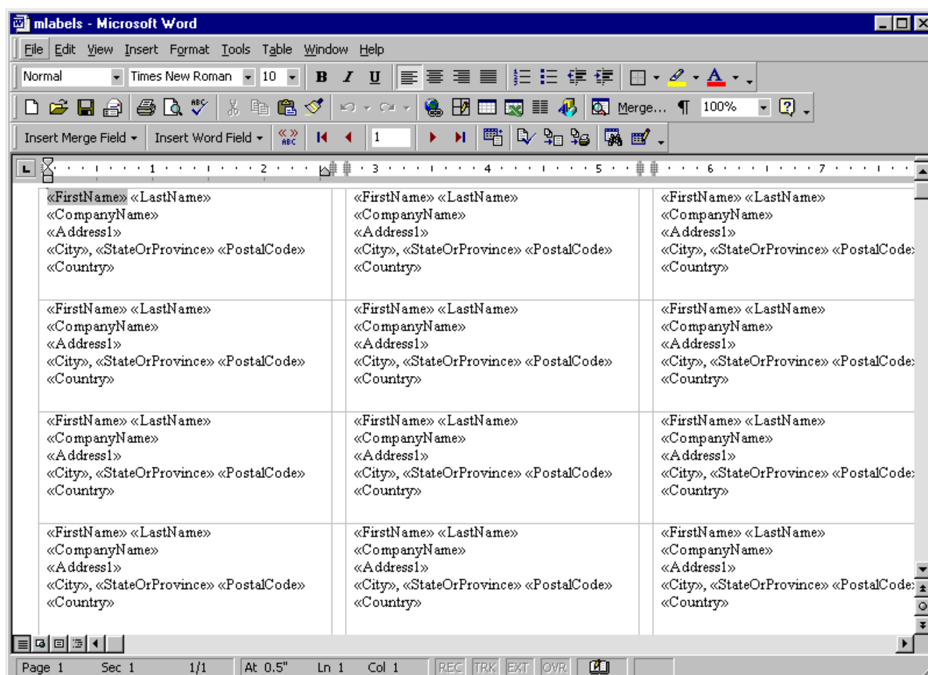
הפקת תוויות מען

תכונת מיזוג הדואר המוכללת ב- Word יכולה לשאוב נתונים מתוך שאילתה או טבלת מסד נתונים של Access, המשמשים כמקור רשומות להפקת תוויות מען. אומנם ניתן למקם באמצעות קוד את שדות מיזוג הדואר, או הסימניות, כך שיתמכו בהצבת נתוני Access במסמכי Word, אולם פעמים רבות קל יותר למקם אותם ידנית במקומות שבהם אתה מבקש להוסיף נתונים במסמך. נוסף לכך, ניתן להיעזר באשפים מוכללים לצורך הפריסה של פקדים בתבניות עם תוויות מרובות בכל עמוד.



תרשים 9.5: טופס זה מפעיל את השיגרה fromAccessToWordTable; כאשר נפסקת הנפשת העבודה במסייע, המיקוד עובר לטבלה שב- Word

בתרשים 9.6 שלהלן מוצג קטע מתוך הקובץ mlabels.doc. שדות מיזוג הדואר מוקמו בעזרת אשף מיזוג הדואר המוכלל, אשר ניתן להפעיל מתפריט **כלים** (Tools). לאחר שמיקמת שפרסם את הפקדים, תוכל לשלוט באמצעות קוד בהדפסת תוויות המבוססות על טבלאות או שאילתות של Access, באמצעות האובייקט MailMerge.



תרשים 9.6: קטע מתוך מסמך Word שנוצר לצורך הדפסת תוויות מען בתבנית Avery 5260

שגרת Word הבאה מיועדת להפעלה מתוך התיקיה ThisDocument של התבנית Normal. היא מתחילה בפתיחת הקובץ mlabels.doc, המכיל את שדות מיזוג הדואר הנראים בתרשים 9.6. לאחר מכן היא מציבה הפניה, myDoc, אל מסמך המבוסס על אותו קובץ. למעשה, הקובץ mlabels.doc אינו מכיל קוד משלו.

```
Sub printPreviewLabels()
Dim myDoc As Document

' Set reference to the document.
    Documents.Open FileName:="mlabels.doc"
    Set myDoc = Documents("mLabels")

' Reference the data source in Access.
    myDoc.MailMerge.OpenDataSource Name:= _
        "C:\Programming Access\Chap09\Chapter9.mdb", _
        Connection:="TABLE WebBasedList"

' Send the labels to a new document.
    With myDoc.MailMerge
        .Destination = wdSendToNewDocument
        .Execute Pause:=True
    End With

' Either Print Preview or Print labels.
    If MsgBox("Do you want to preview before printing?", _
        vbYesNo, "Programming Microsoft Access 2000") = vbYes Then
        ActiveDocument.PrintPreview
    Else
        ActiveDocument.PrintOut
    End If
End Sub
```

המפתח למיזוג דואר עם נתונים מ- Access הוא להפעיל את השיטה OpenDataSource עבור האובייקט MailMerge בתוך מסמך, כמו למשל המסמך שההפניה myDoc מצביעה אליו. פעמים רבות תידרש להקצות שני ארגומנטים, Name (שם) ו- Connection (חיבור). עליך להציב את הארגומנט Name כך שיצביע אל קובץ המקור ב- Access, המכיל את הנתונים למיזוג הדואר. השיגרה printPreviewLables מפנה אל Chapter9.mdb, הקובץ עבור פרק זה בתקליטור הנלווה. השתמש בארגומנט Connection כדי להקצות לאובייקט מסד הנתונים סוג ושם. בעזרת הקוד תוכל להקצות סוג Table או סוג Query. הדוגמה מצביעה אל טבלה ששמה WebBaseList.

השתמש באובייקט MailMerge לכתיבה על גבי התבנית עם שדות מיזוג הדואר, או ליצירת מסמך חדש שיכיל את הנתונים לאחר המיזוג. עליך להשתמש במאפיין Destination של האובייקט MailMerge כדי להקצות את האפשרות שבחרת. בדוגמה

נעשה שימוש בקבוע כדי ליצור מסמך חדש שיכיל את הנתונים לאחר המיזוג. אחרי שתגדיר את כל מאפייני MailMerge הרצויים, החל את השיטה Execute על האובייקט MailMerge. פעולה זאת תגרום למיזוג הנתונים מתוך Access במסמך Word היוצר את תוויות המען. אם אינך צריך עוד את המסמך עם השדות הממוזגים ואת המסמך השולט בפריסת השדות במסמך, סגור את שניהם ידנית, או בעזרת תוכנית אחרת.

ניתן להפעיל בנקל שיגרה כמו printPreviewLabels מתוך Access. השיגרה הבאה עושה זאת בשלוש שורות בלבד. דוגמה זאת מופעלת מתוך מודול סטנדרטי בקובץ Access.

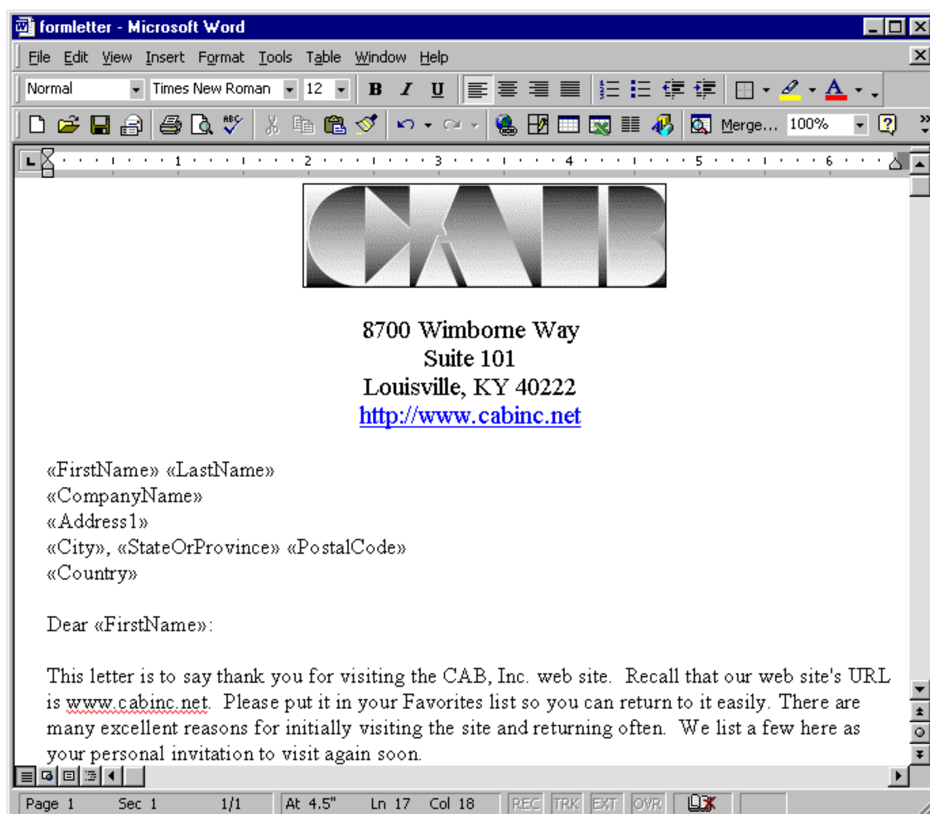
```
Sub runMLabels()
Dim myWDApp As Word.Application
' Open connection to Word and make Word visible.
   Set myWDApp = CreateObject("Word.Application")
   myWDApp.Application.Visible = True
' Run mailing label procedure.
   myWDApp.Application.Run "printPreviewLabels"
End Sub
```

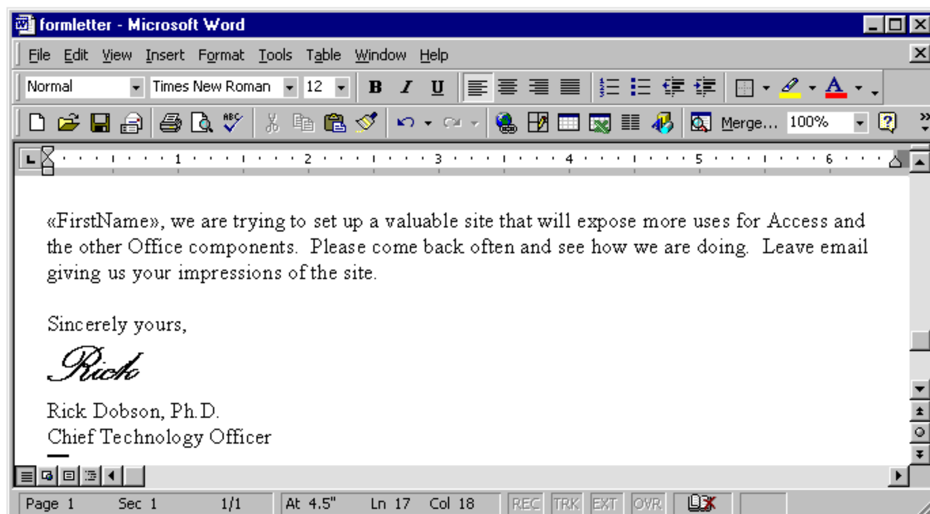
הפקת מכתב אחיד

השיגרה להפקת מכתב אחיד המובאת להלן זהה, בעיקרה, לשיגרה המפיקה תוויות מען, להוציא פריסת שדות מיזוג הדואר, הנעשית במתכונת ישירה יותר. דבר זה נובע מן העובדה שבדרך כלל אין יותר ממופע אחד של כל רשומה בעמוד נתון, בניגוד לרשומות מרובות בכל עמוד. תרשים 9.7 מציג פריסה של מכתב אחיד ב- Word, שהינה זהה לפריסת המכתב שהוצג בפרק 6. אין קושי בהצבת שדה מיזוג דואר בגוף המכתב. דבר זה נראה בלוח התחתון של התרשים.

אל תשתמש באשף מיזוג הדואר אם בכוונתך לשלוט בהדפסת התוויות באמצעות קוד. הקוד שהוא מפיק מכיל פרמטרים רבים נוספים, ומיועד במפורש לשימוש עם האשף, אך לא בהכרח עם יישומים מותאמים אישית. הדוגמה הבאה וכן זאת שבסעיף הקודם מיועדות לשימוש מתוך התיקיה ThisDocument של התבנית Normal. דוגמאות אלה הן נקודות זינוק מצוינות לעיצוב יישומי מכתבים אחידים ותוויות מען משלך. עם זאת, עליך להפעיל את אשף מיזוג הדואר כדי להוסיף בחלון המסמך את סרגל הכלים **מיזוג דואר** (Mail Merge). לאחר שחשפת את סרגל הכלים, תוכל להוסיף שדות מיזוג דואר בכל מקום רצוי במסמך.

השיגרה הבאה פועלת עם המסמך הנראה בתרשים 9.7. תרשים זה מציג קטעים מראש הקובץ formletter.doc ומתחתיתו. השיגרה פותחת קובץ זה. השימוש בשיטה OpenDataSource זהה להפקת מכתב אחיד או תוויות מען. בדוגמה זאת, מוקצים ערכים למאפיינים FirstRecord ו- LastRecord של האובייקט DataSource. השיטה OpenDataSource מציינת פרמטרים המצביעים אל מקור הנתונים שהאובייקט DataSource מייצג. הצבת ערכים במאפיינים FirstRecord ו- LastRecord מגדירה טווח של רשומות שעבור כל אחת מהן יש להדפיס מכתב אחיד. דוגמה זאת פועלת עם סדר ברירת המחדל למיון רשומות במקור הנתונים.





תרשים 9.7: קטעים מתוך מסמך Word המשמש להדפסת מכתבים אחידים

```
Sub printPreviewLetters()
Dim myDoc As Document
' Load file and set a reference to it.
   Documents.Open FileName:="formletter.doc"
   Set myDoc = Documents("FormLetter")
' Reference the data source in Access.
   myDoc.MailMerge.OpenDataSource Name:= _
      "C:\Programming Access\Chap09\Chapter9.mdb", _
      Connection:="TABLE WebBasedList"
' Send the labels to a new document.
   With myDoc.MailMerge
      .Destination = wdSendToNewDocument
      With .DataSource
         .FirstRecord = 5
         .LastRecord = 9
      End With
      .Execute Pause:=True
   End With
' Either Print Preview or Print labels.
   If MsgBox("Do you want to preview before printing?", _
      vbYesNo, "Programming Microsoft Access 2000") = vbYes Then
      ActiveDocument.PrintPreview
   Else
      ActiveDocument.PrintOut
   End If
End Sub
```

ניתן להפעיל את השיגרה מתוך Access על ידי הפעלת השיטה Run עבור אובייקט Application של Word, כפי שמראה השיגרה הבאה. לשם כך דרושות שלוש שורות בלבד. בעוד שהשגרות runFormletters ו- runLabels הן שגרות למודול סטנדרטי ב- Access, תוכל להציב כל אחת מהן, או את שתיהן, מאחורי טופס של Access ולהפעיל אותן באמצעות לחצני פקודה. אם הקוד שלך מציב שאילתה בפרמטר OpenDataSourceConnection, תוכל לשנות את השאילתה בטרם תפעיל את מיזוג הדואר ב- Word, וזאת כדי לשפר את שליטתך בפעולת המיזוג.

```
Sub runFormLetters()
Dim myWDApp As Word.Application
' Open connection to Word and make Word visible.
   Set myWDApp = CreateObject("Word.Application")
   myWDApp.Application.Visible = True
' Run form letter procedure.
   myWDApp.Application.Run "printPreviewLetters"
End Sub
```

10

עבודה עם מסדי נתונים מרובי-משתמשים

בחלק ניכר משארית ספר זה, נעסוק בבניית יישומי Microsoft Access לקבוצות של משתמשים. פרק זה עוסק בכמה סוגיות מרכזיות הנוגעות לריבוי משתמשים: שיתוף קבצים ושליטה באבטחה. תוך סקירת נושא שיתוף הקבצים, הפרק עוסק בבדיקה מפורשת של טכניקות ושאלות בנוגע לשיתוף קבצים ועריכת רשומות. בפרקים הבאים נבחן טכנולוגיות לשיתוף מסדי נתונים של Access, כמו למשל שכפול מסדי נתונים, שימוש בשרת SQL, ושימוש במסדי נתונים של Access באמצעות האינטרנט. בכתיבת פרק זה הנחנו כי יש לך ידע מעשי בסוגיות בסיסיות של ריבוי משתמשים ואבטחה, ולכן התמקדנו בהתפתחויות חדשות בתחום הקוד. לדוגמה, הפרק כולל דוגמאות המשוות את הנעילה החדשה, ברמת השורה, לנעילה ברמת העמוד שהיתה מקובלת עד כה בטיפול בערכות רשומות, וכן דוגמאות המראות כיצד לשלוט באבטחה באמצעות קוד דרך הספריות ADODB ו- ADOX החדשות. הפרק מסתיים בדוגמה של טרנזקציות בסביבה מרובת משתמשים.

שיתוף קבצים

הצעד הראשון בשיתוף יישום של Access הוא הפיכת מצב ברירת המחדל לפתיחה לשיתופי. תוכל לעשות זאת באמצעות הכרטיסיה **מתקדם** (Advanced) בתיבת הדו-שיח **אפשרויות** (Options). כרטיסיה זאת מכילה עוד אפשרויות המקדמות את השיתוף. השיטות SetOption ו- GetOption של האובייקט Application של Access מאפשרות ליישומיך לקבוע הגדרות אלה ולקרוא אותן.

הצב קבצי Access מרובי-משתמשים בתיקיות של ספריות קבצים משותפות. מכיון ש- Access עם מנגנון Jet הוא שרת קבצים ולא מסד נתונים שרת-לקוח, עליך לשאוף לצמצם את גודל הקבצים העוברים ברשת הפיסית. אחת השיטות לעשות זאת היא לפצל יישום לשני קבצים. שמור את טבלאות הנתונים בקובץ Access בתיקיית הספריות המשותפות, והפץ בין שאר המשתמשים קובץ Access אחר, שישמש אותם בתחנות העבודה המקומיות. בקובץ המופץ ייווצר קישור בחזרה אל טבלאות הנתונים שבתיקיות המשותפות. עיצוב מסוג זה מחיש את העבודה ומפחית את נפח התעבורה ברשת. לרוב ניתן לשפר את רמת הביצועים על ידי הכללת עוד תכנים בקובץ המופץ, כמו למשל נתונים שאינם משתנים לעיתים קרובות. תוכל להשתמש בשגרות אוטומטיות לעדכון נתונים מאחסון מקומי בעקבות אירועים קבועים מראש, כמו למשל עם פתיחת היישום, או בתגובה ללחיצת לחצן על ידי המשתמש.



הערה:

תוכל להשתמש באשף פיצול מסדי הנתונים כדי לפצל קובץ Access לשני חלקים (בחר באפשרות **עזרי מסד נתונים** מתוך תפריט **כלים**). אחד הקבצים יכיל את הטבלאות, ואילו האחר יכיל שאילתות, טפסים, דוחות, פקודות מאקרו, מודולים וקיצורי דרך לדפי גישה לנתונים (דפי גישה לנתונים הם אלמנט חדש ב- Access 2000). הצב את הקובץ הראשון בתיקיה משותפת ברשת, וקשר אליו את הקובץ השני. הפץ את הקובץ השני בין המשתמשים בתחנות העבודה השונות.

באפשרותך לקבוע הגדרות אבטחה ברמת המשתמש, כך שמשתמש נתון יוכל לפתוח קובץ מסוים (ומשתמשים אחרים לא יוכלו להשתמש בו). שימוש בלעדי במסד נתונים עשוי להיות נחוץ למשתמש על מנת לבצע שינויים, כמו למשל הוספת מודולים חדשים, או בדיקה חוזרת של מודולים קיימים. בסביבה מרובת משתמשים, הגישה הבלעדית למסד נתונים מתבצעת באמצעות המאפיין Connection Control והתכונה רשימת משתמשים (User List).

המאפיין Connection Control

המאפיין החדש Connection Control מקל על השליטה בנגישות למסדי נתונים באמצעות קוד. ניתן להגדיר את המאפיין הזה עבור האובייקט Connection של ADO (ActiveX Data Objects) כך שלא יתאפשר למשתמשים חדשים להתחבר אל מסד

נתונים, ומשתמשים קיימים לא יורשו להתחבר אליו מחדש לאחר שסגרו את הקובץ. המאפיין Connection Control מאפשר לך להשיג גישה בלעדית אל מסד הנתונים כדי לבצע בו שמירה של מודולים חדשים ושינויים אחרים בעיצובי היישום לאחר שאחרון המשתמשים הנוכחיים ניתק את הקשר עם מסד הנתונים.

שתי השגרות הבאות משנות את הגדרות המאפיין Connection Control. השתמש באוסף Properties של האובייקט Connection כדי להגדיר את המאפיין. ערך של 1 כהגדרת המאפיין גורם לתכונה לסגירה פסיבית כאשר המשתמשים מנתקים את הקשר עם קובץ היישום. ערך של 2 יאפשר למשתמשים אחרים להקים מחדש את הקשר עם מסד הנתונים. תכונה זאת שימושית למפתחים המבקשים להעריך את השינויים שביצעו לאחר ששמרו אותם בקובץ.

```
Sub closeDBConnection()
Dim cnn1 As ADODB.Connection
   Set cnn1 = CurrentProject.Connection
   cnn1.Properties("Jet OLEDB:Connection Control") = 1
End Sub

Sub openDBConnection()
Dim cnn1 As ADODB.Connection
   Set cnn1 = CurrentProject.Connection
   cnn1.Properties("Jet OLEDB:Connection Control") = 2
End Sub
```

רשימת המשתמשים

רשימת המשתמשים, שהוכנסה לשימוש ב- Jet 4, היא ערכה של נתוני משתמשים השמורה בקובץ הנעילה של קובץ mdb.*. נתונים אלה זמינים באופן בלעדי באמצעות ספק Jet, אולם תוכל לקבל גישה אליהם רק באמצעות ADO (אם מסד הנתונים שלך שוכן בקבצים שהשיתוף שלהם הוא לקריאה בלבד, תכונה זאת לא תהיה זמינה, הואיל ו- Jet לא יוצר קובצי נעילה עבור שיתופים לקריאה בלבד. עליך להפעיל את השיטה OpenSchema עבור אובייקט Connection כדי להחזיר ערכת נתונים ובה ארבעה שדות נתונים עבור כל משתמש מן העת האחרונה. השיטה שולפת נתונים מקובץ הנעילה של קובץ mdb.*. ארבעת השדות בערכת הרשומות הם:

- computer_name שם המחשב שממנו מבצע המשתמש את החיבור.
- login_name שם הכניסה לצורך אבטחה ברמת המשתמש.
- connected שדה זה מציין אם המשתמש מחובר כרגע (לפעמים, זיהויי משתמשים נשארים במסד הנתונים הנעול גם לאחר שהמשתמש כבר ניתק את החיבור עם מסד הנתונים של Jet).
- suspected_state שדה זה מציין סיום בלתי-רגיל של החיבור אל מסד הנתונים.

השיגרה הבאה מפעילה את רשימת המשתמשים. צור הפניה אל רשימת המשתמשים בעזרת זיהוי ייחודי גלובלי (Globally Unique ID - GUID) המצביע אל הפקד להחזרת ערכת הרשומות של משתמשים פעילים. הדוגמה מחזירה ערכים לשדות computer_name ו- login_name עבור כל המשתמשים הפעילים. כאשר מניין המשתמשים ברשימה זאת עומד על 1, המפתח יכול לבצע שמירה של שינויים שנערכו במודולים של מסד הנתונים. נוסף לכך, המפתח יכול להשתמש ברשימה כדי לברר אל מי עליו לפנות כדי לקבל שליטה בלעדית במסד הנתונים.

```
Sub listActiveUsers()
Dim cnn1 As ADODB.Connection, rst1 As New ADODB.Recordset
' Set the connection for the current project and invoke the
' OpenSchema method to return the current list of users.
    Set cnn1 = CurrentProject.Connection
    Set rst1 = cnn1.OpenSchema(adSchemaProviderSpecific, , _
        "{947bb102-5d43-11d1-bdbf-00c04fb92675}")

' Print a heading for the User List recordset and enumerate list members.
    Debug.Print "Machine Name    " & "User Name"
    Debug.Print "============" & "    ========="
    Do Until rst1.EOF
        Debug.Print rst1.Fields("computer_name") & rst1.Fields("login_name")
        rst1.MoveNext
    Loop

End Sub
```

שיתוף טפסים

Access מאפשר למשתמשים מרובים לערוך, להוסיף ולמחוק רשומות במסד נתונים, גם ללא קוד. עם זאת, תוכל לשפר כושר זה בסביבה מרובת משתמשים על ידי הפעלת מודולים מאחורי טפסים.

טופס אוטומטי פשוט המבוסס על טבלה או מקור אחר לערכות נתונים עשוי לפעול בסביבה מרובת משתתפים. שני משתמשים, או יותר, יוכלו לפתוח טופס כזה ולדפדף בין הרשומות. כמו כן יוכלו המשתמשים לשנות את ערכת הנתונים המאוגדת של הטופס בכל הדרכים האפשריות ב- Access. ניתן לתכנת את Access ולשנות את ערכי ברירת המחדל השולטים באופן ובעיתוי שבו משתמש אחד יכול לראות את השינויים שהכניס משתמש אחר. קובץ מסד הנתונים המשמש אותנו בפרק זה מכיל חמישה טפסים (WebBaseList, WebBaseList1, WebBaseList2, WebBaseList3 ו- WebBaseList4) המשקפים סוגים שונים של שליטה באמצעות קוד.

נעילת רשומות באופן ידני

בכרטיסיה **מתקדם** (Advanced) בתיבת הדו-שיח **כלים** (Tools), **אפשרויות** (Options), תמצא פקדים לשיתוף טפסים. הלחצן **רשומה בעריכה** (Edited Record) מחיל נעילה פסימית ברמת העמוד. בנעילה מסוג זה, Access נועל את העמוד (או העמודים) המכילים רשומה מרגע שהטופס פותח את הרשומה לצורך עריכה. אם יש בעמוד או בעמודים אלה עוד רשומות, גם אלה ננעלות. תיבת הסימון **פתח מסד נתונים עם נעילה ברמת רשומה** (Open Databases With Record-Level Locking) מחילה נעילה אופטימית על הרשומה הנוכחית בלבד. רשומות אחרות באותו עמוד אינן ננעלות. תכונה זאת, של נעילה ברמת הרשומה, נוצרה לבקשת מפתחים רבים במטרה לצמצם את הסיכוי להתנגשות בגלל טיפול מקביל בסביבה מרובת משתמשים. Microsoft הוסיפה תכונה זאת כדי לבטל את ההשפעה של הגדלת קיבולת העמוד מ- 2KB ל- 4KB (דבר שהיה הכרחי כדי לאפשר עבודה עם תבנית Unicode בשדות המבוססים על תווים במקום בשדות נומריים).

רענון ערכי שדה

בברירת המחדל בכרטיסיה **מתקדם** (Advanced), מרווח הזמן לרענון מציין את מספר השניות המירבי העשוי לחלוף עד שערך שדה שעודכן בטופס מסוים יופיע לעיני משתמשים אחרים. המשתמשים יכולים לרענן שדה נתון באופן ידני על ידי בחירה בפקודה **רענן** (Refresh) מהתפריט **רשומות** (Records), או בכך שיעבירו את הסמן על פני השדה. אם הסמן אינו פעיל בטופס, ייתכן שיחלוף זמן רב יחסית עד שהשינוי יופיע לעיני משתמש אחר המעיין באותה רשומה.

הטופס WebBasedList מציג את התנהגות הגדרות הנעילה והרענון הקיימות בטופס Access בסביבה מרובת משתמשים. אם תפתח שני עותקים שונים של הטופס ותכניס שינויים באותה רשומה באמצעות שני מחשבים שונים (או שתי הפעלות שונות של Access באותו מחשב), תוכל לראות את התנהגות הנעילה ומרווחי הזמן לרענון/עדכון.

הטופס WebBasedList1 הוא גירסה משופרת של הטופס WebBasedList נטול המודולים; הוא משתמש בשתי שיגרות אירועים קצרות, המובאות להלן. שגרת האירוע Form_Open קובעת ערך של 2 שניות באפשרות מרווח זמן לרענון. שגרת האירוע Form_Close מחזירה לתקנו את מרווח זמן ברירת המחדל בן 60 השניות. שים לב, שהשיגרה משתמשת בשיטה SetOption עבור האובייקט Application של Access.

```
Private Sub Form_Open(Cancel As Integer)
   Application.SetOption "Refresh Interval (Sec)", 2
End Sub

Private Sub Form_Close()
   Application.SetOption "Refresh Interval (Sec)", 60
End Sub
```

כאשר שתי הפעלות שונות פותחות את הטופס WebBasedList1, הדבר נראה כאילו עדכוני נתונים פשוטים עוברים מהפעלה אחת למשניה מהר יותר מאשר בטופס WebBasedList. מרווח זמן הרענון הקצר יותר אינו משקף שינויי סדר, הוספה או מחיקה של רשומות בין הפעלות, אולם הוא משקף עדכונים במהירות רבה, וצריכת המשאבים שלו נמוכה.

העדכון הפשוט שמבצעת השיגרה Form_Open עשוי להשפיע באופן דרמטי על ההיענות הנראית לעין של יישום Access לריבוי משתמשים. למעשה, להגדרה זאת אין שום השפעה בעת תפעול בידי משתמש יחיד. מטרתה היחידה היא להשפיע על העיתוי שבו שינויים שעורכים משתמשים אחרים יבואו לידי ביטוי בטפסים (ובגליונות נתונים).

פקדי רענון ושאילתה חוזרת

הטופס WebBasedList2 מכיל לחצן **Refresh** המפעיל את השיטה Refresh עבור הטופס. הקוד שמאחורי אירוע הלחיצה של לחצן זה הוא Me.Refresh. השיטה Refresh פועלת בדיוק כמו פסק הזמן של מרווח הזמן לרענון. גם כאן, המשתמשים אינם רואים פעולות של סדר מיון חדש, הוספת רשומות או מחיקת רשומות, אולם הם רואים מייד כל שינוי שהוכנס ברשומות בידי משתמשים אחרים. למרות שניתן להשיג תוצאות דומות על ידי העברת הסמן על פני השדה, הלחצן נקרא **Refresh** כדי שהמשתמש יידע מייד איזו פעולה הוא מבצע.

השיטה Refresh מהירה, אולם היא אינה מעדכנת טופס כך שישקף רשומה חדשה, והיא מציגה רשומות שנמחקו עם סמן מחיקה. הפעל את השיטה Requery של הטופס כדי להציג את ערכת הרשומות כפי שהיא נראית לאחר שמשתמש אחד, או יותר, הוסיף או מחק רשומות. במצב ברירת המחדל, השיטה Refresh משאירה את הרשומה הנוכחית במצב פעיל, אולם השיטה Requery בוחרת את הרשומה הראשונה לאחר שסיימה את פעולתה. אם תרצה לחזור למיקום הקודם (אשר קרוב לוודאי אינו הרשומה הראשונה), עליך להוסיף לוגיקה מתאימה כדי לנווט בחזרה למיקום הרצוי. אם משתמש מסוים שינה את הקריטריונים למיון, כמו למשל ערכי מפתח ראשי או מפתח למיון, השיטה Requery תשקף את השינויים הללו, אולם היא אינה מעדכנת את סדר הרשומות. השיטה Requery פועלת בסביבת משתמש יחיד ובסביבה מרובת משתמשים כאחת (לשיטה Refresh אין השפעה ביישומים למשתמש בודד).

שתי השגרות הבאות מוסיפות לחצן Requery לצד לחצן Refresh בטופס WebBasedList2. לשיגרה cmdRequery_Click שלושה חלקים. תחילה היא שומרת את ערך המפתח הראשי הנוכחי כדי לסייע באיתור הרשומה מחדש לאחר שהשיטה Requery מעבירה את נקודת ערכת הרשומות לרשומה הראשונה בטופס. שנית, היא מחילה את השיטה Requery. שלישית, היא מעבירה את המיקוד אל שדה המפתח הראשי (ContactID), ומפעילה את הפקודה DoCmdFindRecord. דבר זה מחזיר את הטופס מהרשומה הראשונה אל הרשומה שהיתה הרשומה הנוכחית לפני הפעלת השיטה Requery. אם משתמש אחר מחק את הרשומה הזאת, הטופס יחזיר את התצוגה אל הרשומה הראשונה. תוכל להפעיל לוגיקה משוכללת יותר כדי להציב את הרשומה הנוכחית במיקום אחר.

```
Private Sub cmdRefresh_Click()
   Me.Refresh
End Sub

Private Sub cmdRequery_Click()
Dim pkBefore As Long
' Save primary key value before requery.
   pkBefore = Me.ContactID.Value
' Apply requery method.
   Me.Requery
' Attempt to find primary key value before requery.
   Me.ContactID.SetFocus
   DoCmd.FindRecord pkBefore
End Sub
```

נעילת רשומות באמצעות קוד

נעילת רשומות פירושה נעילת ערכים ברשומה, באופן שמונע ממשתמשים אחרים להכניס בהם שינויים. Access 2000 מאפשר לראשונה נעילת רשומות, במקום נעילת עמודים בלבד. השליטה בעיתוי הנעילה נעשית באמצעות אפשרויות הנעילה בכרטיסיה **מתקדם** (Advanced) שבתיבת הדו-שיח **אפשרויות** (Options). כמו כן ניתן להקצות באמצעות קוד ערכים למאפיין RecordLocks (נעילות רשומה)של טופס נתון, כדי לקבוע אם הנעילה תהיה אופטימית או פסימית, ואם תחול ברמת העמוד או ברמת הטבלה. המאפיין RecordLocks עשוי לקבל אחד משלושה ערכים. הערך 0, שהוא ברירת המחדל, מציין נעילת עמוד אופטימית. הערך 2 מציין נעילת עמוד פסימית. הערך 1 גורם לנעילה ייחודית של הטבלה כולה.

הכרטיסיה **מתקדם** מכילה גם תיבת סימון חדשה לנעילה ברמת הרשומה במקום ברמת העמוד. תיבת סימון זאת נבחרת לפי ברירת המחדל. ההגדרה שהיא מכילה מאפשרת לשני משתמשים לעדכן בו-זמנית שתי רשומות שונות באותו עמוד, באמצעות טפסים. אין אפשרות לנעילה ברמת השורה בנתונים מסוג תזכיר (memo) או בעמודי אינדקס. בדרך כלל, Access אינו תומך במתן שמות משתמשים עם נעילות סותרות. ניתן להגדיר בקוד באופן ידני נעילה ברמת השורה באמצעות השיטה SetOption של האובייקט Application ב- Access. השיטה GetOption מאפשרת החזרת הגדרות לנעילת שורה, לעומת הגדרות לנעילת עמוד.

השגרות Form_Open ו- Form_Close המובאות להלן מטפלות בסגנונות שונים של נעילת עמוד ונעילת רשומה לעומת נעילת עמוד. האירוע Form_Open מאפשר למשתמשי טפסים לבצע נעילה אופטימית או פסימית של עמודים. כאשר המשתמש בוחר באחת האפשרויות, השגרה מציבה ערך False בהגדרת הנעילה ברמת השורה. אם המשתמש לא בחר באחת משתי האפשרויות לנעילת עמוד, השיגרה תציב ערך True בהגדרת הנעילה ברמת השורה. ערכי True ו-False אלה מקבילים לבחירה, או הסרת הבחירה, בתיבת הסימון המייצגת נעילה ברמת הרשומה בכרטיסיה **מתקדם**.

```
Private Sub Form_Open(Cancel As Integer)
' Present a series of message boxes to let a user set the
' page-locking style and record-level locking for a form.
    If MsgBox("Optimistic page locking?", vbYesNo, _
        "Programming Microsoft Access 2000") = vbYes Then
        Me.RecordLocks = 0
        Application.SetOption "Use row-level locking", False
    ElseIf MsgBox("Pessimistic Page locking?", vbYesNo, _
       "Programming Microsoft Access 2000") = vbYes Then
        Me.RecordLocks = 2
        Application.SetOption "Use Row Level Locking", False
    Else
        MsgBox "Setting new row-level locking.", _
            vbInformation, "Programming Microsoft Access 2000"
        Application.SetOption "Use row-level locking", True
    End If
' Handy for confirming selected states
    Debug.Print Me.RecordLocks
    Debug.Print Application.GetOption("Use row-level locking")
End Sub

Private Sub Form_Close()
' Restore default locking.
    Me.RecordLocks = 2
    Application.SetOption "Use row-level locking", True
End Sub
```



הערה:
הגדרות הנעילה שקבע המשתמש הראשון במצב של ריבוי משתמשים, יחולו על כל שאר המשתמשים באותה הפעלה. כאשר אתה בודק שגרות המציבות ערכים באפשרויות הנעילה השונות, עליך לצאת מ- Access ולהפעילו מחדש עבור כל בדיקה כדי שהגדרות הנעילה החדשות יחולו.

לא לכל ההגדרות במאפיין RecordLocks הגדרות מקבילות בכרטיסיה **מתקדם**. שני משפטי ההדפסה שבתחתית האירוע Form_Open משקפים הגדרות נוכחיות לסגנון וסוג נעילת העמוד, וכן אם חלה במערכת נעילה ברמת השורה במקום ברמת העמוד.

Access 2000 מכיל מאפיין אופציונלי להחלפת נעילות מרובות ברמת העמוד בנעילה אחת של הטבלה כולה. עם הגידול במספר הנעילות בכל טבלה, Access יוכל להחליף באופן אוטומטי את נעילות הרשומה הנפרדות בנעילה אחת לכל טבלה. קידום זה בנעילה האוטומטית עשוי להפחית באורח דרמטי את "העלות" של ניהול נעילות, בתוספת היתרון של נעילת רשומות כאשר כמות הנעילות מועטה. תוכל לבחור תכונה זאת על ידי ביטול האפשרות לנעילה ברמת הרשומה כברירת מחדל, ושימוש בשיטה

SetOption עבור האובייקט Application. כמו כן עליך להציב ערך בערך הרישום החדש PagesLockedToTableLock. ערך ברירת המחדל, שהוא 0, הופך ללא זמין את קידום הנעילה האוטומטית. כל ערך גדול מ- 0 מציין את המספר המירבי של נעילות לטבלה בטרם ינסה Jet להחליפן בנעילת טבלה אחת. ערך של 100 פירושו, ש- Jet ינסה להחליף נעילות עמוד בנעילת טבלה בודדת עם הצבתה של הנעילה ה- 101, ה- 201 וכך הלאה, בכל טבלה נתונה. הנסיונות יצליחו רק אם תינתן ל- Jet גישה בלעדית לטבלה (הואיל והצבת נעילה על הטבלה נועלת את הטבלה בפני כל המשתמשים, מלבד בעל הנעילה). הטיפול בנתונים ב- Jet מהיר הרבה יותר כאשר ניתנת לו גישה בלעדית לטבלה. תכונה זאת גם חוסכת מהמפתח את הצורך לקבוע מראש מתי להחיל נעילות טבלה בלעדיות.

שיתוף ערכות רשומות

בערכות רשומות ובמשפטי Jet SQL, משתמש בערכי ברירת מחדל 'חכמים' כדי לקבוע אלו הגדרות נעילת עמוד ונעילת רשומה להחיל. באמצעות קוד ADO המסתמך על סמנים, עוברת ברירת המחדל ב- Jet לנעילה ברמת הרשומה. במשפטי SQL, העשויים בנקל להשפיע על אלפי שורות, Jet עובר לנעילה ברמת העמוד כברירת המחדל. דבר זה מאפשר ליישום שלך ליהנות מקידום נעילה (אם אפשרות זאת זמינה).

אם אתה מחיל נעילה ברמת השורה על ערכות רשומות, תוכל להגיע לשיפור בהתאמה וברמת הביצועים רק אם תקפיד במפורש על גלישת משימות תחזוקה של ערכות רשומות בטרנזקציות. השתמש בשיטות BeginTrans, CommitTrans ו- Rollback (שנדון בהן בהמשך פרק זה) כדי להבטיח גלישה בכל פעולת רשומה. בלא אמצעי זה, יצטברו עוד ועוד נעילות, עד שיגיעו לערך של הגדרת הרישום FlushTransactionTimeout.

אם אתה מחיל נעילה ברמת הרשומה, עליך לשקול בזהירות אלו סוגי נתונים להקצות לשדות ואלו אורכי רשומה לקבוע, כדי שתוכל לקיים בקרה על גודל קובץ מסד הנתונים. נעילה ברמת הרשומה גורמת לכתיבת שורות במקום עמודים; שורות המשתרעות מעבר לגבול העמוד יש להעביר לעמוד חדש. באמצעות סוג נתונים בעל אורך קבוע, תוכל לקבוע במדויק את אורכן של רשומות; לא ניתן לעשות זאת כאשר טבלאות הנתונים ב- Access מכילות תווים בעלי אורך משתנה (עם זאת, שפת הגדרת הנתונים ב- Jet SQL מכילה סוג נתונים CHAR, המאפשר לך לקבוע את אורכם של שדות מסוג 'מחרוזת'. דבר זה עשוי, בתורו, לסייע בבקרה על גידול הנפח של רשומות כתוצאה מרשומות שעמוד יחיד אינו יכול להכילן).

נעילה ברמת השורה

השיגרה הבאה מראה כיצד מופעלת נעילת שורות ביישום מרובה משתמשים. השיגרה מדמה יישום לשני משתמשים על ידי הפעלת שני חיבורים כנגד אותו קובץ נתונים. ליישום דרוש קובץ נוסף, ששמו TestRowLock.mdb. הפעל את השיגרה במצב **צעד** כדי לראות כיצד נעילת השורות מצליחה, או נכשלת, בהתאם לנסיבות השונות.

```
Sub RowLockingPessimistic()
' Use TestRowLocking.mdb to run this demo of  row locking;
' DO NOT HAVE TESTROWLOCK.MDB OPEN WHEN RUNNING THE DEMO.
Dim cnn1 As New ADODB.Connection
Dim cnn2 As New ADODB.Connection
Dim rs As New ADODB.Recordset
Dim rs2 As New ADODB.Recordset
Dim j As Integer
On Error GoTo ErrHandler
' Open two connections against the same database, both using row locking mode;
' Jet OLEDB:Database Locking Mode info - 0 is "page mode", 1 is "row mode"
' Open first connection to TestRowLocking.mdb.
cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data " & _
    "Source=c:\Programming Access\Chap10\TestRowLocking.mdb;" & _
    "Jet OLEDB:Database Locking Mode=1;"
' Open second connection to TestRowLocking.mdb.
cnn2.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
    "Data Source=c:\Programming Access\Chap10\TestRowLocking.mdb;" & _
    "Jet OLEDB:Database Locking Mode=1;"
' Open a recordset in the first connection and begin editing the first row.
rs.Open "TestRowLocking", cnn1, adOpenKeyset, _
    adLockPessimistic, adCmdTableDirect
' Edit first row field value.
Debug.Print rs.Fields("col1")
rs.Fields("col1") = 2
' rs2.Update; putting an update here would close the record lock
Debug.Print rs.Fields("col1")
' Open a recordset in the second connection.
rs2.Open "TestRowLocking", cnn2, adOpenKeyset, _
    adLockPessimistic, adCmdTableDirect
' Attempt to edit the first row; even in row locking, since we are in the pessimistic
' locking mode, this will fail.
Debug.Print rs2.Fields("col1")
rs2.Fields("col1") = 3
Debug.Print rs2.Fields("col1")
' Move to another row in the same page.
rs2.MoveNext
' Attempt to edit the next row; this works under the row locking mode because
' the lock only applies to a single row, not the whole page.
Debug.Print rs2.Fields("col1")
' Should not fail (row locking)
rs2.Fields("col1") = 4
Debug.Print rs2.Fields("col1")
```

```
' Update recordsets.
rs.Update
rs2.Update
' Close connections and exit.
cnn1.Close
cnn2.Close
Set cnn1 = Nothing
Set cnn2 = Nothing

Exit Sub

ErrHandler:
For j = 0 To cnn1.Errors.Count - 1
    Debug.Print "Errors from cnn1 connection"
    Debug.Print "Conn Err Num : "; cnn1.Errors(j).Number
    Debug.Print "Conn Err Desc: "; cnn1.Errors(j).Description
Next j
For j = 0 To cnn2.Errors.Count - 1
    Debug.Print "Errors from cnn2 connection"
    Debug.Print "Conn Err Num : "; cnn2.Errors(j).Number
    Debug.Print "Conn Err Desc: "; cnn2.Errors(j).Description
Next j
Resume Next
End Sub
```

שים לב שהניסיון שנעשה בחיבור השני לכתוב ברשומה הראשונה של הקובץ TestRowLock.mdb נכשל, הואיל והרשומה עדיין פתוחה בחיבור הראשון. ניסיון זה עשוי להצליח אם החיבור הראשון יסיר את הנעילה מהרשומה (למשל, על ידי הפעלת משפט rs.Update). נסיונו של החיבור השני לכתוב ברשומה השנייה מצליח מייד, מכיון שלא חלה כל נעילה על רשומה זאת. הלוגיקה לטיפול בשגיאות מציגה את הערכים מהאוסף Errors בחיבור של Jet.

נעילה ברמת עמוד

להלן קטע מהשיגרה ()PageLocking בעיצוב דומה, העושה שימוש בנעילת עמוד במקום בנעילת רשומה. שים לב שהערך של **Database Locking Mode**, שהיה 1 בדוגמה הקודמת, התחלף ב- 0 בדוגמה זאת. ערכים אלה מציינים, בהתאמה, מעקב ברמת השורה וברמת העמוד. שני נסיונותיו של החיבור השני לעדכן רשומות נכשלים, הואיל והחיבור הראשון פתח נעילה, ועמוד יחיד מכיל את כל הרשומות של טבלה קצרה זאת.

```
' Open first connection to TestRowLocking.mdb.
cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data " & _
    "Source=c:\Programming Access\Chap10\TestRowLocking.mdb;" & _
    "Jet OLEDB:Database Locking Mode=0;"
' Open second connection to TestRowLocking.mdb.
cnn2.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
    "Data Source=c:\Programming Access\Chap10\TestRowLocking.mdb;" & _
    "Jet OLEDB:Database Locking Mode=0;"
' Open a recordset in the first connection and begin editing the first row.
rs.Open "TestRowLocking", cnn1, adOpenKeyset, _
    adLockPessimistic, adCmdTableDirect
' Edit first row field value.
Debug.Print rs.Fields("col1")
rs.Fields("col1") = 2
Debug.Print rs.Fields("col1")
' Open a recordset in the second connection.
rs2.Open "TestRowLocking", cnn2, adOpenKeyset, _
    adLockPessimistic, adCmdTableDirect
' Attempt to edit the first row; because of pessimistic page locking mode
' setting, this fails.
Debug.Print rs2.Fields("col1")
rs2.Fields("col1") = 3
Debug.Print rs2.Fields("col1")
' Move to another row in the same page.
rs2.MoveNext
' Attempt to edit the next row; this fails too because the next row is on the
' same page.
Debug.Print rs2.Fields("col1")
 rs2.Fields("col1") = 4
Debug.Print rs2.Fields("col1")
' Update recordsets.
rs.Update
rs2.Update
```

אבטחה

Access מכיל מערך מגוון של תכונות אבטחה, התומכות בצרכים השונים של יישומי Access השונים. רוב יישומי Access למשתמשים מרובים עשויים להפיק תועלת באבטחה ברמת המשתמש, המאפשרת למפתחים ליצור קבוצות של משתמשים. אלא שביישומים מסוימים יש צרכים מיוחדים נוספים. בסעיף זה נביא סקירה קצרה על טכניקות אבטחה נוספות, מעבר לאבטחה ברמת המשתמש. הסעיפים הבאים עוסקים בגישות שונות לניהול אבטחה ברמת המשתמש באמצעות ADO, דרך הקוד.

חלופות לאבטחה ברמת-משתמש

אחד היתרונות של Access הוא יכולתו לשרת קהלי יעד שונים. יישומים מסוימים הם עתירי-קוד, על כן עליך להבטיח את השקעתך בקוד מקור. יישומים אחרים משרתים קבוצות עבודה קטנות עם כשרים טכניים מוגבלים, אך עדיין נדרשת אבטחה ברמה מינימלית כדי למנוע גישה לנתונים. ביישומים אחרים, היתרון נובע מהפעלת ממשק משתמש מותאם אישית המגביל את הפונקציונליות על ידי חשיפת סדרה מוגבלת של פקודות.

שימוש בממשק מותאם אישית

לעיתים ניתן ליצור אבטחה מספקת ליישום על ידי החלפת ממשק Access הרגיל בממשק מותאם אישית. בתפריט **כלים** (Tools), בחר באפשרות **הפעלה** (Startup) כדי לפתוח תיבת דו-שיח המאפשרת לך לציין כותרת וסמל ליישום מותאם אישית, טופס פתיחה מותאם אישית, ותפריטים מותאמים אישית שיחליפו את אלה המקובלים ב-Access. תיבת דו-שיח זאת מאפשרת גם להשבית את החלון **מסד נתונים** (Database) ואת שורת המצב. כמו כן תוכל לטפל באמצעות קוד בתכונות של תיבת הדו-שיח **הפעלה**. אם סוג זה של טיפול מתאים לצורכי האבטחה שלך, עליך לשקול אם כדאי לתגבר אותו בשיטות Show ו- Hide, ובמאפיינים Visible ו- Hidden לאובייקטים בחלון **מסד נתונים** (עיין בפרקים קודמים, שם תמצא פרטים על טכניקות ליצירת תכונות מותאמות אישית: פרק 8 מתאר כיצד לתכנת את תבנית האובייקט CommandBars כדי לבנות תפריטים מותאמים אישית, ולהתאים תפריטים מוכללים; פרק 5 מראה כיצד לבנות טופס פתיחה מותאם אישית).

הגדרת סיסמה למסד נתונים

באפשרותך לחייב משתמשים להזין סיסמה כדי לקבל גישה בלתי מוגבלת לכל אובייקטי הנתונים ומסדי הנתונים ב- Access. בהשוואה לאבטחה ברמת המשתמש, אין קושי בניהול סיסמאות. אבטחה באמצעות סיסמה היא אמצעי הולם כאשר פועלת במשרד קבוצה שלחבריה דרושה גישה זהה לכל האלמנטים של קובץ מסד נתונים, אולם לא כל עובדי המשרד חברים בקבוצה.

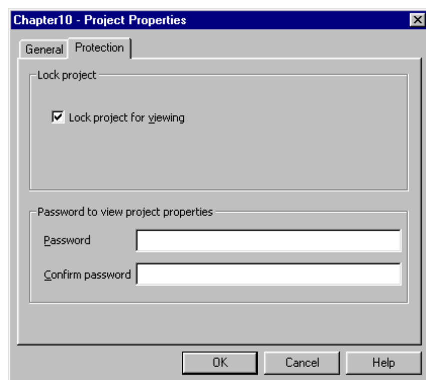
לא ניתן להשתמש בקובץ מאובטח בסיסמה כחבר בקבוצת חברים משוכפלים, מכיון שב-Jet, שכפול מסד נתונים אינו יכול ליצור סינכרון עם קובץ מאובטח בסיסמה (ראה פרטים בפרק 11). כמו כן עליך להיזהר ביצירת קישור עם קבצי מסד נתונים מאובטחים בסיסמה מכיון שכל אדם המסוגל לגשת אל הקובץ המקשר אל הקובץ המוגן זוכה בגישה בלתי מוגבלת לקובץ המוגן. יתר על כן, Access שומר גירסה מוצפנת של הסיסמה לצד נתונים אחרים אודות הקובץ המקושר. לבסוף, אם מישהו יחליף את הסיסמה עבור קובץ מקושר, Access ידרוש את הסיסמה החדשה בפעם הבאה שקובץ מסד נתונים אחר יקושר אליו.

כדי להקצות ולבטל סיסמה למסד נתונים, דרושה לך גישה בלעדית לקובץ. בצע את הפעולות הבאות:

1. כדי להקצות סיסמה לקובץ, פתח את הקובץ על ידי בחירת האפשרות **פתח בלעדית** (open Exclusive) בלחצן **פתח** (Open) שבתיבת הדו-שיח **פתיחה**.
2. בתפריט **כלים**, בחר **אבטחה** (Security), **הגדרת סיסמת מסד נתונים** (Set Database Password).
3. בתיבת הדו שיח **קביעת סיסמת מסד נתונים**, הזן את הסיסמה שבחרת בתיבות הטקסט **סיסמה** ו**אמת סיסמה**, ולחץ על **אישור**. בפעם הבאה שמשתמש יפתח את הקובץ, היישום ידרוש ממנו את הסיסמה.
4. לאחר שביצעת פתיחה בלעדית של מסד נתונים, בחר **כלים**, **אבטחה**, **בטל קביעת סיסמת מסד נתונים** (Unset Database Password). הסר את הסיסמה על ידי הקלדת הסיסמה בתיבת הדו-שיח **בטל קביעת סיסמת מסד נתונים**. פעולה זאת מבטלת את הדרישה הראשונית לסיסמה כתנאי לגישה אל מסד הנתונים.

הגדרת סיסמה למודול

ב- Access 2000 ניתן להחליף אבטחה ברמת המשתמש באבטחה באמצעות סיסמת מודול. גישה חדשה זאת יוצרת אחידות בין Access לבין שאר מרכיבי Office 2000, וחלה על כל מודולי המחלקה העצמאיים והמודולים הסטנדרטיים, וכן על המודולים שמאחורי טפסים ודוחות.



תרשים 10.1: השתמש בכרטיסיה **Protection** בתיבת הדו-שיח **Project Properties** כדי להגדיר סיסמת אבטחה למודולים של פרויקט

הגדרת סיסמת האבטחה לכל המודולים בפרויקט VBA נעשית בו-זמנית באמצעות עורך Visual Basic (VBE). בתפריט **Tools** (כלים) של הפרויקט, בחר בפקודה **Properties** (מאפיינים) כדי להציג את תיבת הדו-שיח **Project Properties** (מאפייני פרויקט). הכרטיסיה **Protection** (הנראית בתרשים 10.1) מכילה את תיבת הסימון **Lock Project For Viewing** (נעל פרויקט להצגה), ותיבות טקסט להזנה ואישור של סיסמה למודול. הקצאת סיסמה כתנאי להצגת מודולים בפרויקט אינה מספיקה כדי למנוע מהקוד לפעול כאילו אינו מוגן. אם הקצית סיסמה אך לא בחרת בתיבת הסימון **Lock Project For Viewing**, כל משתמש יוכל לערוך את הקוד, ורק תיבת הדו-שיח

Project Properties תהיה מוגנת. לביטול סיסמת האבטחה של מודולים בפרויקט, מחק את כל הערכים בכרטיסיה **Protection**.

לאחר אבטחת המודולים בסיסמה, עליך להכניס את הסיסמה פעם אחת נוספת כדי שתוכל לצפות, לערוך או להכניס קוד חדש. תוכל לאבטח טפסים ודוחות בעזרת אבטחת רמת משתמש (user-level security) או סיסמת מודול (module password). אבטחת רמת משתמש מיושמת לעיצוב ושימוש בטפסים ודוחות. תוכל לחייב את המשתמש לקבל רשות Modify Design כדי להוסיף פקדים לטפסים. המשתמש גם יצטרך את הסיסמה למודולים בפרויקט, כדי לכתוב פרוצדורות לאירועים עבור הפקד. ידיעת הסיסמה למודולים בפרויקט אינה מאפשרת להוסיף פקדים או להסיר פקדים מטופס. בנוסף, הרשות Modify Design (שינוי עיצוב) אינה מאפשרת למשתמש לשנות את המאפיין HasModule של טפסים ודוחות ל-NO. המשתמש צריך להכניס תחילה את הסיסמה עבור המודולים של הפרויקט.

שימוש בקבצי mde.*

קובץ mde.* מעניק אבטחה מוחלטת לקוד של קובץ מסד נתונים ב- Access. בהמרת קובץ mdb.* לקובץ mde.*, Access מהדר את כל המודולים, מסלק קוד ניתן לעריכה, ודוחס את הקובץ החדש תוך שימור קובץ mdb.* המקורי. מסד הנתונים שלך יתכווץ בגודלו בגלל הסרת הקוד הניתן לעריכה. כמו כן, מכיון שההמרה יוצרת מיטוב בניצולת הזיכרון, מהירות הפעולה של הקוד תגדל.

כדי לבצע המרה של קובץ mdb.* לקובץ mde.*, עליך לקבל גישה בלעדית לקובץ (עיין בסעיף שכותרתו "הגדרת סיסמה למסד נתונים", לעיל). בתפריט **כלים** בחר באפשרות **עזרי מסד נתונים** (Database Utilities), ולאחר מכן בחר באפשרות **יצירת קובץ MDE** (Make MDE File). לאחר שמירת קובץ ההמרה, הקפד לשמור גם את הקובץ המקורי. בהגנה מסוג זה, השיטה היחידה לערוך את הקוד בקובץ מסד נתונים או להוסיף עליו, היא לערוך את השינויים בקובץ המקורי, ולאחר מכן להמיר אותו בקובץ mde.*.

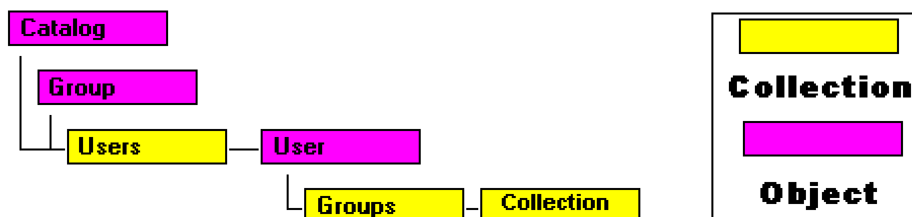
בקובץ mde.* חלות מספר מגבלות:

- לא ניתן להכניס שינוי בטפסים, דוחות או מודולים, או להוסיף חדשים.
- לא ניתן לייבא או לייצא טפסים, דוחות או מודולים אל תוך קובץ mdb.* רגיל. עם זאת, ניתן לייבא ולייצא ללא קושי טבלאות, שאילתות, פקודות מאקרו וקיצורי דרך אל עמודי גישה לנתונים במסדי נתונים אחרים.
- לא ניתן להוסיף, למחוק או לשנות הפניות לספריות אובייקטים או מסדי נתונים אחרים.
- לא ניתן לבצע שינויי קוד דינמיים, הואיל וקבצי mde.* אינם מכילים קוד ניתן לעריכה (בפרק 7 תמצא דוגמה המציגה מגבלה זאת).
- לא ניתן לבצע המרה של חבר קיים בקבוצת עותקים משוכפלים לקובץ mde.*, אולם קובץ mde.* יכול להשתתף בקבוצת עותקים משוכפלים.

➢ קובץ ‎*.mde‎ יכול להפנות לקובץ מסד נתונים אחר רק אם אותו קובץ אף הוא מסוג ‎*.mde‎. עליך להמיר קבצי ‎*.mdb‎ (או קבצי תוספות ‎*.mda‎) בטרם תמיר את קובץ ‎*.mdb‎ המכיל את ההפניה אליהם. על ההפניה החדשה להצביע אל קובץ ‎*.mde‎ החדש.

בקרה על אבטחה ברמת-משתמש באמצעות קוד

באבטחה ברמת המשתמש ניתן להגדיר קבוצת עבודה המורכבת מחשבונות משתמש ומחשבונות קבוצה. ניתן ליצור באמצעות קוד חשבונות משתמש וחשבונות קבוצה, וכן להקצות הרשאות לחשבונות אלה. מבנה ADOX החדש תומך בפונקציונליות מסוג זה באמצעות האובייקט Catalog, וכן אובייקטי האוסף Users ו- Groups. תרשים 10.2 מציג את ההיררכיה: קבוצות יכולות להשתייך למשתמשים, ומשתמשים יכולים להשתייך לקבוצות; הן משתמשים והן קבוצות שייכים לאובייקט Catalog.



תרשים 10.2: יחסי האובייקטים User ו- Group אל האובייקט Catalog

תוכל להקצות הרשאות למשתמשים, לקבוצות, או לשניהם גם יחד. ככלל, השיטה היעילה ביותר היא להקצות הרשאות לקבוצות. עליך למחוק את כל הרשאות ברירת המחדל מחשבונות המשתמשים הנפרדים, ולאחר מכן להציב משתמשים בכל הקבוצות המתאימות. בסוג זה של עיצוב ברמת המשתמש, ניתן לנהל הרשאות על ידי הצבת משתמשים בקבוצות, והקצאת הרשאות לקבוצות, מכיון שכל הרשאות הקבוצה חלות על המשתמשים החברים בה. על ידי הגבלת ההרשאות המוקצות לקבוצות, עיצוב זה יוצר נקודה מרכזית אחת לניהול הרשאות.

הדוגמאות הבאות מראות כיצד ליצור באמצעות קוד משימות ניהול טיפוסיות ברמת המשתמש. הואיל ומטרתן העיקרית היא להדגיש שגרות בסיסיות בניהול אבטחה, רק שתיים מהן מכילות לוגיקה ללכידת שגיאות.

חיבור אל מסד נתונים מאובטח

השיגרה הבאה יוצרת חיבור אל מסד נתונים מאובטח. מחרוזת החיבור כוללת ארבעה משפטים. המשפט הראשון מייעד את המאפיין Provider לאובייקט Connection של השיגרה. הוא מציין את Jet 4 כספק OLE DB. המשפט השני מקצה לחיבור את מאפיין מסד הנתונים של המערכת, כלומר השם והנתיב של קובץ נתוני קבוצת העבודה (הקובץ systemdemo.mdw בתיקיית Office; עליך להעתיק את הקובץ באופן ידני אל

תוך תיקיית Office). המשפט השלישי מצין את מקור הנתונים, שמבקרה זה הוא קובץ מסד הנתונים המאובטח UserLevel.mdb. המשפט הרביעי מציין ערכי זיהוי משתמש וסיסמה לצורך כניסה אל מסד הנתונים המאובטח. בדוגמה זאת, השיגרה נכנסת למערכת בשם Admin. אם לא תשנה את הגדרות ברירת המחדל, יוקנו למשתמש זה הרשאות ניהול מיוחדות.

```
' Turn logon procedure on before running this procedure.
' Assign password of "password" to Admin user account.

Sub openUserLevel()
Dim cnn1 As New ADODB.Connection
Dim rst1 As New ADODB.Recordset
' Open connection to target user-level secured data source; specify path for
workgroup information file; designate logon ID and password.
    cnn1.Provider = "Microsoft.Jet.OLEDB.4.0"
    cnn1.Properties("Jet OLEDB:System database") = _
        "C:\Program Files\Microsoft Office\Office\systemdemo.mdw"
    cnn1.Open "Data Source=C:\Programming Access\Chap10\UserLevel.mdb;" & _
        "User Id=Admin;Password=password;"
' Print first field from first record to confirm connection.
    rst1.Open "WebBasedList", cnn1, , , adCmdTable
    Debug.Print rst1.Fields(0)
    cnn1.Close
End Sub
```

שתי השורות שלאחר ההערה פותחות ערכת רשומות המבוססת על החיבור ומדפיסות את ערך השדה הראשון מתוך הרשומה הראשונה. דבר זה נועד לאימות פעולתה התקינה של הדוגמה. הטבלה WebBasedList היא אותה טבלה ששימשה בדוגמה לריבוי משתמשים, לעיל.

כדי שהשיגרה OpenUserLevel תפעל, עליך להפעיל את שגרת הכניסה למערכת. דבר זה כרוך במתן סיסמה למשתמש Admin. כמו כן דרוש לשיגרה גם קובץ נתוני קבוצת עבודה. במקרה זה, על שמו להיות systemdemo.mdw, ועליך לשמור אותו בנתיב שצוין בשיגרה. התקליטור הנלווה כולל הן את קובץ מסד הנתונים המאובטח, והן את קובץ נתוני קבוצת העבודה, לביצוע קל ופשוט של בדיקת השיגרה.

הוספה ומחיקה של משתמשים

כאשר מפתחים ומנהלים יישום מותאם אישית עם אבטחה ברמת המשתמש, מתעורר בדרך כלל צורך בהוספה ובמחיקה של משתמשים. בטרם תוכל להוסיף משתמשים, עליך להיכנס למערכת כחבר בקבוצה Admins, למשל Admin. תוכל להשתמש בשיטה Append של האוסף Users כדי להוסיף משתמשים לקטלוג או לקבוצה. עליך לציין עבור המשתמש החדש שם, ובאפשרותך גם להקצות סיסמה. ADO מאפשר הקצאה של

סיסמה בשלב מאוחר יותר בעזרת השיטה ChangePassword של האובייקט User. למרבה הצער, אינך יכול להקצות PID, כיון שערך זה נבחר באופן אקראי על ידי ADO.

שתי השגרות הבאות מציגות אחת מהגישות להפעלת השיטה Append כדי להוסיף משתמש חדש ביישום. השיגרה callMakeUser מפעילה את השיגרה MakeUser ומעבירה שני ארגומנטים. הארגומנט הראשון מקצה שם משתמש חדש, השני שולח סיסמה. בדוגמה, המחרוזת "password" מכילה את הערך של ארגומנט הסיסמה.

```
' Make sure NewUser account does not exist prior to running this procedure;
' for example, run callDeleteUser first.

Sub callMakeUser()
   makeUser "NewUser", "password"
End Sub

Sub makeUser(usrName As String, secureWord As String)
Dim cat1 As New ADOX.Catalog
  cat1.ActiveConnection = _
       "Provider=Microsoft.Jet.OLEDB.4.0;" & _
       "Data Source=C:\Programming Access\Chap10\UserLevel.mdb;" & _
       "Jet OLEDB:System database=C:\Program Files\" & _
       "Microsoft Office\Office\systemdemo.mdw;" & _
       "User Id=Admin;Password=password;"
   cat1.Users.Append usrName, secureWord
End Sub
```

השיגרה makeUser מגדירה יעד לקבוצה החדשה בעזרת ההגדרה ActiveConnection באובייקט Catalog. שים לב, שהיא מקצה זיהוי משתמש עם הסמכות ליצור משתמש חדש, ומצביעה אל קובץ נתוני קבוצת עבודה. השיטה Append בשיגרה makeUser מוסיפה חבר חדש באובייקט Catalog. לכן, המשתמש החדש עדיין אינו חבר באף אחת מהקבוצות. באפשרותך גם להוסיף חבר לאובייקט Group, כך שלמשתמש תוקנה חברות מיידית באותה קבוצה. באחת מהדוגמאות הבאות ננקטת טכניקה זאת.

שתי השגרות הבאות מוחקות משתמש מהקטלוג של מסד נתונים. התחביר בשיטה Delete של האוסף Users זהה לתחביר בשיטה Delete של אובייקטי האוסף Tables, Procedures ו- Views. השיגרה הראשונה, callDeleteUser, מעבירה ארגומנט אחד – שם המשתמש – אל השיגרה השנייה, deleteUser. השיגרה השנייה מוחקת את המשתמש מהקטלוג, ובמקביל מוחקת את המשתמש גם מכל שאר הקבוצות.

```
' Make sure NewUser account exists prior to running this procedure;
' for example, run callMakeUser.

Sub callDeleteUser()
   deleteUser "NewUser"
End Sub
```

```
Sub deleteUser(usrName As String)
Dim cat1 As New ADOX.Catalog
    cat1.ActiveConnection = _
        "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Programming Access\Chap10\UserLevel.mdb;" & _
        "Jet OLEDB:System database=C:\Program Files\" & _
        "Microsoft Office\Office\systemdemo.mdw;" & _
        "User Id=Admin;Password=password;"
    cat1.Users.Delete usrName
End Sub
```

כדי למחוק משתמש, עליך להיכנס למסד הנתונים כחבר בקבוצה Admins. להפעלת השיטה Delete לא נדרשת סיסמה. כל שדרוש לשיגרה השנייה הוא ארגומנט מחרוזת המכיל את שם המשתמש שיש למחוק.

הקצאת קבוצות למשתמשים

אחת הטכניקות המקובלות לניהול הרשאות היא הקצאת קבוצות למשתמשים, וניהול ההרשאות עבור אותן קבוצות. ההרשאות של משתמשים נגזרות כולן מהחברות בקבוצה. הדוגמאות בסעיף זה מוסיפות ומוחקות חברות בקבוצה מתוך חשבון משתמש. בשתי הדוגמאות נעשה שימוש בקבוצה המוכללת Users, אולם אותן טכניקות יפות גם לקבוצות מותאמות אישית.

שתי השגרות הבאות מוסיפות קבוצה לחשבון משתמש ששמו NewUser. הקפד לוודא שחשבון המשתמש אכן קיים בטרם תפעיל את השיגרה. השיגרה הראשונה, callAddGroupToUser, מעבירה שם משתמש ושם קבוצה אל השיגרה הבאה. השיגרה AddGroupToUser, המשתמשת בשיטה Append כדי להוסיף את האובייקט Group לאוסף Groups עבור המשתמש. הדוגמה מעבירה ארגומנטים לשיגרה השנייה, וארגומנטים אלה מורים לה להוסיף את הקבוצה Users למשתמש NewUser.

```
Sub callAddGroupToUser()
   AddGroupToUser "NewUser", "Users"
End Sub

Sub AddGroupToUser(usrName As String, grpName As String)
On Error GoTo AddTrap
Dim cat1 As New ADOX.Catalog
Const acctNameAlreadyExist = -2147467259
    cat1.ActiveConnection = _
        "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Programming Access\Chap10\UserLevel.mdb;" & _
        "Jet OLEDB:System database=C:\Program Files\" & _
        "Microsoft Office\Office\systemdemo.mdw;" & _
        "User Id=Admin;Password=password;"
```

```
    cat1.Groups.Append grpName
    cat1.Users(usrName).Groups.Append grpName

AddExit:
    Exit Sub

AddTrap:
    If Err.Number = acctNameAlreadyExist Then
        Resume Next
    Else
        Debug.Print Err.Number; Err.Description
    End If
End Sub
```

השיגרה השנייה מפעילה את השיטה Append בניסיון ליצור קבוצה עם השם של הארגומנט השני שהועבר אליה. כשמדובר בקבוצות, שיגרה זאת פועלת בין אם הקבוצה כבר קיימת ובין אם לאו. הואיל ו- Users הוא חשבון קבוצה מובנה, הוא ממילא קיים תמיד. אם לא קיימת עדיין קבוצה עם שם הארגומנט השני, השיטה Append תצליח; אם לא, השיגרה תיפול לתוך מלכודת שגיאות ותחולל את מספר השגיאה 2147467259-. לאחר מכן היא תמשיך אל המשפט הבא. בשלב הבא השיגרה תוסיף את הקבוצה לאוסף Groups עבור האובייקט NewUser. גם כאן, אם הקבוצה כבר קיימת באוסף Groups של המשתמש, השיגרה תמשיך אל המשפט הבא.

שתי השגרות הבאות מוחקות קבוצה מתוך אוסף Groups של משתמש. השיגרה הראשונה, callRemoveUserFromGroup מעבירה שם משתמש ושם קבוצה כפרמטרים אל השיגרה השנייה, removeUserFromGroup, וזו מבצעת את המשימה. הואיל ואין בדוגמה זאת בדיקה לאיתור שגיאות, עליך לוודא שהקבוצה שייכת למשתמש. תוכל לעשות זאת על ידי הפעלת הדוגמה הקודמת.

```
' Make sure the group account exist for the user prior to running this procedure;
'for example, run callAddGroupToUser.

Sub callRemoveUserFromGroup()
    removeUserFromGroup "NewUser", "Users"
End Sub

Sub removeUserFromGroup(usrName As String, grpName As String)
Dim cat1 As New ADOX.Catalog
    cat1.ActiveConnection = _
        "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=C:\Programming Access\Chap10\UserLevel.mdb;" & _
        "Jet OLEDB:System database=C:\Program Files\" & _
        "Microsoft Office\Office\systemdemo.mdw;" & _
        "User Id=Admin;Password=password;"

    cat1.Users(usrName).Groups.Delete grpName
End Sub
```

הפעל את השיטה Delete כדי למחוק קבוצה מהאוסף Groups של אובייקט User. שים לב למיפרט ההירכי עבור משתמש יחיד. לאחר זיהוי המשתמש, התחביר מחייב ציון של האוסף Groups, ולבסוף, של השיטה Delete. התחביר מציין את שם הקבוצה כפרמטר לשיטה Delete.

יצירה, מחיקה ומעקב אחר קבוצות בקטלוג

אם בכוונתך לפתח פתרונות מותאמים אישית ברמת המשתמש, בוודאי תרצה ליצור קבוצות מותאמות אישית בעלות שמות שמשמעותם מובנת ללקוחותיך, ושההרשאות המוקנות להן תואמות את הדרישות המיוחדות של היישום המותאם אישית שפיתחת. ארבע הדוגמאות הבאות מבצעות, בהתאמה, את הדברים הבאים: יצירת קבוצה מותאמת אישית, מחיקת קבוצה מותאמת אישית, הכנת דוח ובו פירוט כל הקבוצות בקטלוג נתון והקבוצות המשויכות לכל חשבון משתמש, והחלפת מצב החברות של קבוצה באוסף Users.

שתי השגרות הבאות מוסיפות קבוצה ששמה MySecretGroup1. לאחר הפניה אל קובץ מסד נתונים עם זיהוי משתמש המספיק לביצוע ההוספה, השיגרה מפעילה את השיטה Append של האוסף Groups. חובה לציין גורם מכיל עבור האוסף. כאשר מוסיפים קבוצה חדשה לאוסף Users של הפרויקט, הגורם המכיל הוא אובייקט Catalog. כאשר אתה מציב קבוצה באוסף Groups של אובייקט User, עליך לציין את המשתמש כאובייקט הבסיס עבור האוסף Groups.

```
' Make sure MySecretGroup1 does not exist before running this procedure;
' for example, run callDeleteGroup.

Sub callMakeGroup()
   makeGroup "MySecretGroup1"
End Sub

Sub makeGroup(grpName As String)
Dim cat1 As New ADOX.Catalog
   cat1.ActiveConnection = _
      "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=C:\Programming Access\Chap10\UserLevel.mdb;" & _
      "Jet OLEDB:System database=C:\Program Files\" & _
      "Microsoft Office\Office\systemdemo.mdw;" & _
      "User Id=Admin;Password=password;"

   cat1.Groups.Append grpName
End Sub
```

שתי השגרות שלהלן מוחקות קבוצה מקטלוג. בטרם תפעיל את השגרות, עליך לוודא שהקבוצה כבר קיימת בקטלוג. ברר זאת על ידי הפעלת הדוגמה הקודמת. למעשה, הדוגמה הבאה מוחקת את הקבוצה שהתווספה בדוגמה הקודמת.

```
' Make sure MySecretGroup1 exists prior to running this procedure;
' for example, run callMakeGroup.

Sub callDeleteGroup()
   deleteGroup "MySecretGroup1"
End Sub

Sub deleteGroup(grpName As String)
Dim cat1 As New ADOX.Catalog
   cat1.ActiveConnection = _
      "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=C:\Programming Access\Chap10\UserLevel.mdb;" & _
      "Jet OLEDB:System database=C:\Program Files\" & _
      "Microsoft Office\Office\systemdemo.mdw;" & _
      "User Id=Admin;Password=password;"
   cat1.Groups.Delete grpName
End Sub
```

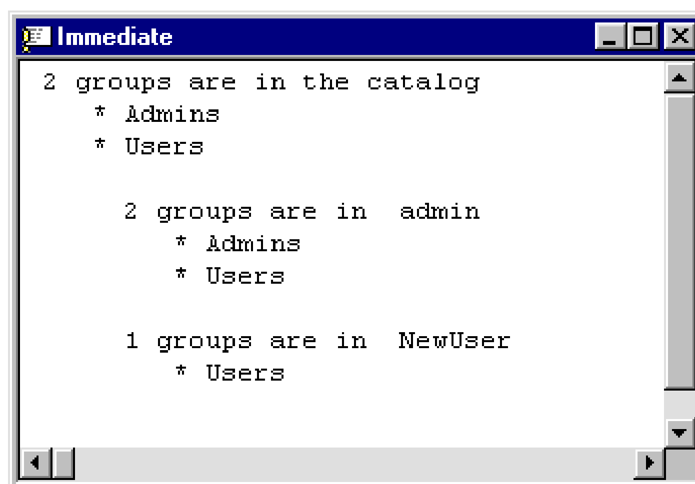
התחביר למחיקת קבוצה דומה מאוד לתחביר המשמש להוספת קבוצה. הוא מפעיל את השיטה Delete של האוסף Groups בקטלוג. לשיטה מועבר פרמטר יחיד – שם הקבוצה המיועדת למחיקה.

תוך כדי הוספה ומחיקה של קבוצות ומשתמשים, והקצאה מחדש של קבוצות למשתמשים, תוכל בנקל ליצור דוח מותאם אישית העוקב אחר החברויות בקבוצות עבור האובייקט Catalog ואובייקטי User הנפרדים. השיגרה הבאה מפרטת את הקבוצות באובייקט Catalog שמצביע אל מסד נתונים מוגדר. לאחר מכן היא מפרטת את חברי האוסף Groups עבור כל משתמש באוסף Users של הקטלוג.

```
Sub listGroupsInCat()
Dim cat1 As New ADOX.Catalog
Dim grp1 As New ADOX.Group, usr1 As New ADOX.User
cat1.ActiveConnection = _
   "Provider=Microsoft.Jet.OLEDB.4.0;" & _
   "Data Source=C:\Programming Access\Chap10\UserLevel.mdb;" & _
   "Jet OLEDB:System database=C:\Program Files\Microsoft Office\" & _
   "Office\systemdemo.mdw;" & _
   "User Id=Admin;Password=password;"
' Groups in overall Catalog
   Debug.Print cat1.Groups.Count & " groups are in the catalog"
   For Each grp1 In cat1.Groups
      Debug.Print String(3, " ") & "* " & grp1.Name
   Next grp1
   Debug.Print
```

```
' Groups in each user
'Creator and Engine are special users that appear in the
'Users collection, but do not need to be tracked.
    For Each usr1 In cat1.Users
        If usr1.Name <> "Creator" And usr1.Name <> "Engine" Then
            Debug.Print String(5, " ") & usr1.Groups.Count & _
                " groups are in  " & usr1.Name
            For Each grp1 In cat1.Users(usr1.Name).Groups
                Debug.Print String(8, " ") & "* " & grp1.Name
            Next grp1
            Debug.Print
        End If
    Next usr1
End Sub
```

בראש כל פירוט של חברי האוסף Groups השיגרה מדווחת על מספר החברים באוסף על ידי הפניה אל המאפיין Counts עבור האוסף. שים לב, בתרשים 10.2 להלן, שדבר זה משתנה בהתאם למשתמש. המשתמש Admin משתייך לקבוצות המוכללות Admins ו- Users. תוכל לנצל את הדוגמאות הקודמות ליצירה ומחיקה של משתמשים, קבוצות וחברויות של משתמשים בקבוצות.



תרשים 10.3: דוח על חברות בקבוצה מתוך השיגרה listGroupsInCat

השיגרה הבאה מדגימה את אחד השימושים האפשריים של השיגרה listGroupsInCat מהדוגמה הקודמת. השיגרה toggleNewUserInAdminsGroup עושה בדיוק מה שמשתמע משמה. היא מחליפה את מצב החברות של האובייקט NewUser בקבוצה Admins. כמו כן היא מתעדת את המצב הנוכחי של האובייקט NewUser בקבוצה Admins על ידי הפעלת השיגרה listGroupsInCat.

```
Sub toggleNewUserInAdminsGroup()
On Error GoTo ToggleTrap
Dim cat1 As New ADOX.Catalog
Const notInCollection = 3265
cat1.ActiveConnection = _
    "Provider=Microsoft.Jet.OLEDB.4.0;" & _
    "Data Source=C:\Programming Access\Chap10\UserLevel.mdb;" & _
    "Jet OLEDB:System database=C:\Program Files\Microsoft Office\" & _
    "Office\systemdemo.mdw; User Id=Admin;Password=password;"
cat1.Users("NewUser").Groups.Delete ("Admins")

ToggleExit:
    listGroupsInCat
    Exit Sub

ToggleTrap:
    If Err.Number = notInCollection Then
        cat1.Users("NewUser").Groups.Append "Admins"
    Else
        Debug.Print Err.Number; Err.Description
    End If
    Resume Next
End Sub
```

שים לב שתהליך שינוי המצב מסתמך על לכידת שגיאות. לאחר שהיא מתחברת למסד הנתונים הרצוי ולקובץ נתוני קבוצת העבודה באמצעות ההפניה באובייקט cat1, השיגרה מנסה למחוק את Admins מתוך האוסף Groups ב- NewUser. אם הניסיון עולה יפה, השיגרה מפעילה את השיגרה listGroupsInCat, ומסתיימת. אם לא, מתחוללת שגיאה. אם מתרחשת שגיאה מכיון שהקבוצה אינה נמצאת באוסף Groups של המשתמש, השיגרה מוסיפה את Admins לאוסף Groups של האובייקט NewUser. לאחר מכן השיגרה ממשיכה כאילו לא אירעה שגיאה, ומסתיימת.

הגדרת הרשאות

השתמש בשיטה SetPermissions עבור האובייקטים Group ו- User כדי לנהל את ההרשאות המוקנות לחשבון מאובטח. הפעל את השיטה GetPermissions עבור אובייקטים אלה, כדי להחזיר ערך ארוך המציין את סוגי ההרשאות המוקצים לקבוצה או למשתמש. שתי השיטות מאפשרות מיגוון רחב של תוצאות: הן יכולות להקצות ולדווח על הרשאות שונות עבור מספר סוגי אובייקטים של מסדי נתונים. נוסף לכך, תוכל להשתמש בשיטה SetPermission כדי להקצות, לבטל ולמנוע הרשאות, ולבקר את אופן השימוש בהן.

שתי השגרות הבאות מעניקות לקבוצה הרשאות מלאות לכל טבלה חדשה. הגדרת ההרשאה עבור טבלה חדשה אינה משפיעה על טבלאות קיימות. לכן ייתכן מצב שבו לקבוצה יש הרשאות מלאות לכל הטבלאות החדשות, אך אין לה כלל הרשאות לטבלאות הקיימות.

```
' Make sure MySecretGroup1 exists before running procedure.

Sub callSetAllTablePermissionsForGroup()
   setAllTablePermissionsForGroup "MySecretGroup1"
End Sub

Sub setAllTablePermissionsForGroup(grpName As String)
Dim cat1 As New ADOX.Catalog
Dim grp1 As New ADOX.Group, usr1 As New ADOX.User
cat1.ActiveConnection = _
   "Provider=Microsoft.Jet.OLEDB.4.0;" & _
   "Data Source=C:\Programming Access\Chap10\UserLevel.mdb;" & _
   "Jet OLEDB:System database=C:\Program Files\Microsoft Office\" & _
   "Office\systemdemo.mdw;" & _
   "User Id=Admin;Password=password;"
cat1.Groups(grpName).SetPermissions Null, adPermObjTable, _
   adAccessSet, adRightFull
End Sub
```

השיגרה הראשונה מעבירה שם קבוצה, MySecretGroup1 אל השיגרה השנייה. השיגרה השנייה מפעילה את השיטה SetPermissions עבור חבר הקבוצה הנושא שם זה. לכן עליך לוודא שהקבוצה קיימת בטרם תפעיל את השיגרה או להוסיף לוגיקה ללכידת שגיאות. הפרמטר הראשון בשיטה מכיל ערך Null מפורש. בדרך כלל, פרמטר זה מציין שם של אובייקט מסד נתונים, כמו למשל, טבלה. ערך Null מציין שברצונך להגדיר הרשאות עבור אובייקטי מסד נתונים חדשים. הפרמטר השני קובע את סוג האובייקט, במקרה זה Table. הפרמטר השלישי משמש כפועל; הוא מציין כי הפקודה תגדיר הרשאה. קבועים נוספים מציינים פעולות שונות, כמו למשל ביטול הרשאות, אשר השיטה יכולה לנקוט. הפרמטר הרביעי מעניק למשתמש זכויות מלאות. השיטה והפרמטרים שלה מעניקים לקבוצה MySecretGroup1 זכויות מלאות בכל הטבלאות החדשות בקובץ מסד הנתונים UserLevel.mdb עם קובץ נתוני קבוצת העבודה systemdemo.mdw.

עיצוב זה הוא עיצוב בסיסי וגמיש, העשוי לשמש במיגוון של נסיבות שונות. לדוגמה, כדי לבטל את כל הזכויות בטבלאות חדשות, עליך להחליף בשיטה SetPermissions את הפרמטר השלישי, adAccessSet, ב- adAccessRevoke. כדי להגדיר זכויות עבור אובייקט מסד נתונים קיים, עליך להחליף, בפרמטר הראשון, את הערך Null בשמו של אובייקט מסד הנתונים.

כיצד לחבר בין כל המרכיבים

שתי השגרות הבאות שואבות מחתך של הדוגמאות הקודמות ומציגות פן נוסף בשיטה SetPermissions. השיגרה הראשונה מפעילה את השיגרה makeGroup כדי ליצור קבוצה חדשה בקובץ נתוני קבוצת העבודה systemdemo.mdw. לאחר מכן היא מפעילה את השיגרה השנייה ומעבירה אליה את שם הקבוצה החדשה, וכן את שם אובייקט מסד הנתונים שעבורו היא מבקשת להקצות הרשאות. שתי השורות האחרונות בשיגרה הראשונה יוצרות משתמש חדש ששמו NewUser2, ומוסיפות את הקבוצה MySecretGroup2 לאוסף Groups שלו. בדרך זאת חלות על המשתמש NewUser2 כל ההרשאות שהוקצו לקבוצה MySecretGroup2 באמצעות השיגרה השנייה.

```
Sub callSetRIDTablePermissionsForGroupTable()
' This procedure makes a group called MySecretGroup2 and assigns
' Read/Insert/Delete Permissions for WebBasedList table to MySecretGroup2.
Then,
' it creates NewUser2 and assigns MySecretGroup2 to NewUser2. Before running
' this, delete MySecretGroup2 and NewUser2 from UserLevel.mdb if they exist.
makeGroup "MySecretGroup2"
setRIDTablePermissionsForGroupTable "MySecretGroup2", "WebBasedList"
makeUser "NewUser2"
AddGroupToUser "NewUser2", "MySecretGroup2"
End Sub

Sub setRIDTablePermissionsForGroupTable(grpName As String, tblName)
Dim cat1 As New ADOX.Catalog
Dim grp1 As New ADOX.Group, usr1 As New ADOX.User
cat1.ActiveConnection = _
   "Provider=Microsoft.Jet.OLEDB.4.0;" & _
   "Data Source=C:\Programming Access\Chap10\UserLevel.mdb;" & _
   "Jet OLEDB:System database=C:\Program Files\Microsoft Office\" & _
   "Office\systemdemo.mdw;" & _
   "User Id=Admin;Password=password;"
cat1.Groups(grpName).SetPermissions tblName, adPermObjTable, adAccessSet, _
   adRightRead Or adRightInsert Or adRightDelete
End Sub
```

השיגרה השנייה מקצה לקבוצה MySecretGroup2 הרשאות קריאה, הוספה ומחיקה עבור הטבלה WebBasedList שבקובץ UserLevel.mdb. שיגרה זאת דומה לדוגמה קודמת שהתאימה להענקת זכויות לאובייקט מסד נתונים מוגדר, אולם שיגרה זאת משרשרת שלוש זכויות נפרדות כדי לקבל ערכה משולבת של הרשאות. שים לב שהתחביר משרשר זכויות בעזרת האופרטור Or.

טרנזקציות

באפשרותך לארוז באמצעות טרנזקציות שתי פעולות מסד נתונים, או יותר, כך שכולן ייצאו אל הפועל, או שאף אחת מהן לא תתבצע. טרנזקציות הן דבר שימושי במצבים משני סוגים: כאשר מוטל על היישום שלך לבצע סדרה של פעולות כיחידה (כמו למשל העברת כספים בין חשבון חיסכון לחשבון עובר ושב); וכן כאשר עליך להאיץ קבוצה של פעולות מסד נתונים באמצעות גלישה בתוך טרנזקציה (מכיון שכתיבה אל זיכרון מטמון זמני מהירה יותר מכתיבה אל דיסק עבור כל אלמנט בקבוצת הטרנזקציה).

שלושה אובייקטי Connection ב- ADO תומכים בטרנזקציות: BeginTrans, CommitTrans ו- Rollback. האובייקט BeginTrans פותח רמת טרנזקציה. Jet תומך בחמש טרנזקציות מקוננות, לכל היותר. לאחר שהפעלת טרנזקציה, באפשרותך לסיים אותה בעזרת השיטה CommitTrans או בעזרת השיטה Rollback. השתמש בשיטה CommitTrans כדי להשלים את פעולות מסד הנתונים. הפעל את השיטה Rollback כדי לסיים טרנזקציה מבלי לבצע את פעולות מסד הנתונים שהופעלו מאז שיטת BeginTrans האחרונה.

השיגרה הבאה מפעילה שלוש שיטות טרנזקציה עבור האובייקט Connection. היא עוברת בלולאה על פני הרשומות במסד נתונים, ומחפשת התאמה לאחד משלושה קריטריונים. כאשר השיגרה מוצאת התאמה, היא כותבת Us בשדה ש-Access מאחסן במטמון נתונים, במקום בקובץ מסד הנתונים. השיגרה מפעילה את השיטה BeginTrans ממש לפני הפעלת לולאת Do. בסוף הלולאה השיגרה שואלת את המשתמש אם ברצונו לבצע את השינויים. אם המשתמש משיב בחיוב, הדוגמה מפעילה את השיטה CommitTrans כדי לבצע את השינויים במסד הנתונים. אם המשתמש משיב בשלילה, השיגרה מוחקת את השינויים על ידי הפעלת השיטה Rollback.

```
Sub changeInTrans()
On Error GoTo changeTrap
Dim cnn1 As ADODB.Connection
Dim rst1 As New ADODB.Recordset
Dim iChanges As Long
Const conLockedByAnotherUser = -2147217887
' Open recordset based on WebBasedList in current project.
Set cnn1 = CurrentProject.Connection
rst1.Open "WebBasedList", cnn1, adOpenKeyset, adLockPessimistic, adCmdTable

' Loop through all records to find those to update to US.
cnn1.BeginTrans
Do Until rst1.EOF
    If rst1.Fields("Country") = "" _
        Or IsNull(rst1.Fields("Country")) = True _
        Or rst1.Fields("Country") = "USA" Then
```

```
        rst1.Fields("Country") = "US"
        iChanges = iChanges + 1
    End If
rst1.MoveNext
Loop

' Commit all changes if user says so.
' Roll changes back otherwise.
If MsgBox("Are you sure that you want to update" & _
    " these " & iChanges & " records?", vbYesNo, _
    "Programming Microsoft Access 2000") = vbYes Then
    cnn1.CommitTrans
Else
    cnn1.RollbackTrans
End If
Exit Sub

changeExit:
    Exit Sub
changeTrap:
    If Err.Number = conLockedByAnotherUser Then
      MsgBox "Recordset not available for update. Try again later.", vbCritical, _
            "Programming Microsoft Access 2000"
    Else
      Debug.Print Err.Number; Err.Description
    End If
    cnn1.RollbackTrans
    Resume changeExit

End Sub
```

השיגרה changeInTrans מפעילה את השיטה Rollback גם אם מתרחשת שגיאה בזמן ריצה, כמו למשל כאשר מסד הנתונים ננעל על ידי משתמש אחר. במקרה כזה, לא מתאפשר לערוך רשומה אחת, או יותר. במסדי נתונים מסוימים, פעולה המתבצעת בקבוצת משנה של רשומות עלולה להשחית את מסד הנתונים כולו. אם השיגרה נתקלת בשגיאה, מכל סיבה-שהיא, היא מבטלת כל פעולה תלויה ועומדת לפני שהיא מסתיימת. הפעלת השיטה Rollback שומרת על תקינותו של קובץ מסד הנתונים.

11

שכפול מסדי נתונים

יישומי Microsoft Access מאפשרים להציג לפני קבוצות משתמשים שונות עותקים נפרדים, או שכפולי מסדי נתונים, תוך ניהול כל השכפולים בתור מסד נתונים אחד. אפשרות זאת חשובה במיוחד ביישומי Access מתקדמים למשתמשים מרובים, או ביישומים למשתמשים מועטים הזקוקים לגישה בלעדית למסד הנתונים, בעוד שאחרים קוראים או מעדכנים את תכולתו. שכפול של מסדי נתונים יעיל וחשוב בפרויקטים שבהם נדרש לסניפים או לעובדים ניידים שיתוף חלקי במסד נתונים, כאשר הגירסה המלאה שמורה במטה הפרויקט. ניתן לשתף נתונים מבלי לפתוח חיבור ברציפות בין שתי נקודות.

מכיון ש- Access 2000 הוא הגירסה השלישית של Access הכוללת את האפשרות לשכפול מסדי נתונים, תכונה זאת כבר שוכללה ושולבה בצורה טובה. פרק זה כולל סקירה כללית על נושא השכפול ומציג את תכונות השכפול החדשות ב- Access 2000. כמו כן הוא מכיל הסברים על ניהול השכפול בעזרת ספריית JRO 2.1. דוגמאות הקוד ישרתו היטב את מי שרק החל לרכוש בקיאות בתכנות פתרונות שכפול, ויסייעו למי שכבר בקיא בתכנות שכפולים ב- DAO (Data Access Objects) לעבור ל- ADO (ActiveX Data Objects). הדיון בנושא JRO במסגרת פרק זה הורחב וכולל גם סוגיות אשר אינן קשורות במישרין לשכפול.

כיצד עובד השכפול

Access מאפשר שכפול עותקים מרובים של מסד נתונים באמצעות רשת תקשורת מקומית (LAN), רשת תקשורת מרחבית (WAN), רשת אינטרא-נט ארגונית, רשת אקסטרא-נט או האינטרנט. המשתמש יכול להכניס שינויים בעותק הנפרד שלו, לפי צרכיו. החיבור בין העותקים עשוי להיות בלתי פעיל פרקי זמן ארוכים. כמו כן ניתן לקבוע ששינויים המוכנסים בעותק מסוים ישוכפלו ויופיעו גם ביתר העותקים.

השם הקיבוצי של קבוצת עותקי מסד נתונים ב- Access הוא **קבוצת עותקים משוכפלים** (replica set); ניתן לעבוד בו-זמנית עם שתי קבוצות כאלה, או יותר. כל עותק בקבוצה נקרא שכפול, ועל אחד העותקים בקבוצה לשמש **תבנית בסיס לעיצוב** (design master). תבנית הבסיס לעיצוב נבדלת מיתר השכפולים בכך שהיא מסוגלת להעביר שינויים מבניים (כמו למשל דוחות וטפסים חדשים) ושינויים בנתונים, אל שאר חברי קבוצת העותקים המשוכפלים (מוצרים אחרים המאפשרים שכפול מסדי נתונים אמנם מכילים תמיכה לחילופי נתונים בין העותקים, אולם אינם מאפשרים הפצה של שינויים מבניים בין חברי קבוצת עותקים משוכפלים).

שכפול מסדי נתונים מתרחש בין זוגות של שכפולים. השינויים מופצים בין חברי קבוצת עותקים משוכפלים תוך כדי תהליך הסינכרון של העותקים זה עם זה. כל עותק בזוג יכול לשלוח את השינויים שהוכנסו בו אל בן זוגו. אם אין התנגשות בין השינויים, העותקים מעדכנים את עצמם בחבילות השינויים שהם מקבלים זה מזה. אם קיימת התנגשות בשינוי מסוים מכיון ששני העותקים שינו את התכולה באותה נקודה, אחד השינויים 'ינצח' והשני 'יפסיד'. Access שומר את השינוי 'המפסיד' בטבלת התנגשויות. תוכל להשתמש בטבלה זאת כדי לפתור באופן ידני את ההתנגשויות, או שהיישום יוכל להחיל כללים אוטומטיים לעיבוד ההתנגשויות. Access מכיל אשף לפתרון התנגשויות; תוכל לבנות אשף נפרד שיחליף או ישלים את פעולתו של אשף זה.

שגיאות שכפול עלולות להתרחש כאשר שינוי מסוים הוא חוקי בעותק מקומי, אולם אינו חוקי באחד, או יותר, מהשכפולים האחרים בתוך קבוצה, כמו לדוגמה הזנת רשומה עם מפתח ראשי זהה בשני עותקים נפרדים בקבוצת עותקים משוכפלים. כאשר מתבצע סינכרון בין שני העותקים, מנגנון מסד הנתונים Jet דוחה את הרשומה עם המפתח הראשי הכפול. גם בתנאים אחרים עשויות להתרחש שגיאות שכפול, כמו למשל החלת כלל לאימות שדה בעותק מסוים, אשר כתוצאה ממנו נוצרים נתונים בלתי חוקיים בעותק אחר.

שוני עשוי להיווצר בין שכפולים באחת משתי דרכים. ראשית, ייתכן שהם יצטרכו להשלים מחזור סינכרון שבמסגרתו כל השכפולים יחליפו נתונים זה עם זה. ניתן לפתור בעיה זאת על ידי השלמת פעולות הסינכרון בין חברי קבוצת העותקים המשוכפלים. שונות עלולה להיווצר גם כתוצאה מהתנגשויות ושגיאות שכפול – גם אם יש סינכרון מלא בין השכפולים.

ניתן לשכפל מסדי נתונים בחמש דרכים:

- בעזרת הסמל **המזוודה** (Briefcase)
- בעזרת הפקודה **שכפול** (Replication)
- בעזרת תכנות ב- JRO (או ב- DAO)
- בעזרת **Replication Manager** (מנהל השכפול)
- בעזרת סינכרון דרך האינטרנט

שכפול בעזרת סמל המזוודה

בעזרת סמל **המזוודה** (Briefcase) שעל שולחן העבודה של Windows, יכול משתמש קצה לקחת עימו עותק של מסד נתונים מחוץ לאתר ולבצע בו עדכונים. עם החזרתו, ניתן לסנכרן את השכפול שבמחשב הנישא עם העותק שבמחשב האישי או המחלקתי. מפתחים יכולים להשתמש בשכפול בעזרת סמל המזוודה כדי להציב תבנית בסיס לעיצוב במחשב הנישא, ולבנות בעזרתו דוחות וטפסים מותאמים אישית. לאחר מכן, הם יכולים לסנכרן אובייקטי מסד נתונים חדשים עם אלה שבעותק העבודה השוטף.

סמל המזוודה של Microsoft זמין בשולחן העבודה של Microsoft Windows ב-Windows 9x וב- Microsoft Windows NT 4 (ייתכן שהמזוודה אינה מותקנת במחשב שלך המופעל במערכת הפעלה Windows 9x. כדי להתקין את המזוודה, לחץ לחיצה כפולה על האפשרות **הוספה/הסרה של תוכניות** (Add/Remove Program) שבלוח הבקרה. לחץ על הכרטיסיה **התקנת Windows** (Windows Setup) שבתיבת הדו-שיח **מאפייני הוספה/הסרה של תוכניות** (Add/Remove Program Properties), בחר באפשרות **עזרים** (Accessories) ולחץ על הלחצן **פרטים** (Details). בתיבת הדו-שיח **עזרים**, בחר באפשרות **מזוודה**, לחץ על **אישור** ופעל לפי ההנחיות). גרירה של קובץ mdb.* מה**סייר** (Explorer) אל סמל המזוודה גורמת להמרת מסד נתונים מתבנית סטנדרטית לתבנית המצוידת בטבלאות ובשדות מיוחדים, התומכים בשכפול (ביישומי Microsoft Office אחרים, המזוודה רק יוצרת עותקים של קבצים שלמים מבלי לבצע סינכרון בין העותקים). המפשר (reconciler) של המזוודה מותיר את המקור המעודכן כתבנית בסיס לעיצוב באתר המקור, ומציב עותק משוכפל במזוודה. תוכל לשנות זאת אם דרושה לך תבנית בסיס לעיצוב במזוודה.



הערה:

כאשר המפשר (reconciler) של המזוודה יוצר קבוצת עותקים משוכפלים, הוא שואל אם ברצונך לשמור גרסת גיבוי של מסד הנתונים המקורי. בחר באפשרות זאת, אלא אם כן אתה בטוח שלא תצטרך לחזור אל גרסת המקור, הואיל ותהליך ההמרה מוסיף טבלאות ושדות חדשים רבים. אין שיטה פשוטה ואוטומטית לסלק את הטבלאות והשדות הללו בעזרת כלי Access מוכללים. לפחות ספק חיצוני אחד (www.trigeminal.com) מציע תוכנית עזר שמסירה שדות של מערכת שכפול מטבלאות המשתמש.

פקודת השכפול

בחר **כלים** (Tools), **שכפול** (Replication) כדי לקבל גישה לפקודות להפיכת מסד נתונים לבר-שכפול, ליצירת עותקים משוכפלים נוספים, לסינכרון בין עותקים, לפתרון התנגשויות וליצירת גרסאות דגם ראשוני של יישומי שכפול. יצירת עותק עם התכונה החדשה **מנע מחיקות** (Prevent Deletes) אפשרית רק דרך תפריט **כלים** (אם כי ניתן לשלוט בו באמצעות קוד לאחר שנוצר).

כדי לפתור התנגשויות ושגיאות שכפול, תיאלץ קרוב לוודאי לבצע סקירה ידנית של החריגים הנפרדים המופיעים בטבלאות ההתנגשויות. גם אם בסופו של דבר תתכנת כללי פשרה מותאמים אישית, תצטרך לבצע הערכה ידנית של הכללים הללו בטרם תאמץ אותם בעבודה השוטפת.

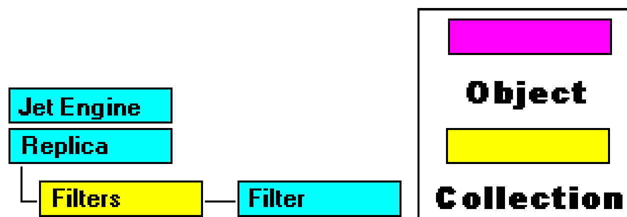
תכנות JRO

בעיקרו של דבר, תכנות JRO אינו אלא גישת ADO, הואיל והוא מסתמך על אובייקטי קישור של ADO. בעוד ש- ADO היא טכנולוגיה אוניברסלית לגישה לנתונים, הרחבת JRO פועלת אך ורק עם מנגנון מסדי הנתונים Jet. אם תכנתת פתרונות שכפול מותאמים אישית ב- Access 95 או ב- Access 97, זה הזמן המתאים לעבור מתכנות ב- DAO לתכנות JRO.

מבנה JRO תומך בשלוש משימות כלליות:

- יצירה וניהול של קבוצות עותקים משוכפלים
- דחיסה והצפנה של מסדי נתונים
- רענון מטמון הזיכרון כדי לשפר את רמת הביצועים הנראית

שלושת האובייקטים המרכזיים בתבנית JRO מוצגים בתרשים 11.1. הראשון הוא האובייקט JetEngine, התומך בתכונות שהן ייחודיות למנגנון מסדי הנתונים Jet, לרבות דחיסה והצפנה של מסדי נתונים, ורענון מטמון הזיכרון.



תרשים 11.1: מבנה JRO

האובייקט המרכזי השני הוא Replica. אובייקט זה מייצג עותק משוכפל של מסד נתונים. אובייקטי Replica הם אבני היסוד בבניית קבוצת עותקים משוכפלים. ניתן לנהל קבוצה כזאת על ידי טיפול במאפיינים ובשיטות של אובייקטי Replica. להלן חלק מהפונקציות שבאפשרותך לנהל:

- הפיכת מסד נתונים לבר-שכפול
- יצירת עותקים מתבנית הבסיס לעיצוב ומעותקים אחרים
- סינכרון בין עותקים
- קריאת אפשרות השכפול של אובייקטים נפרדים

- קביעת תבנית בסיס חדשה לעיצוב
- שליטה באורך תקופת השמירה של היסטוריית שכפול

מבנה JRO מכיל גם מאפיינים לניהול תכונות חדשות ומתוקנות, כמו למשל חשיפה (visibility), סוג שכפול ופתרון התנגשויות המבוסס על קדימויות.

האובייקט המרכזי השלישי הוא Filter. השתמש באובייקט זה להגבלת התכולה בעותקים חלקיים של מסדי נתונים. ניתן לבסס מסנן על טבלה או על יחס. JRO כולל אוסף Filters עבור כל המסננים של עותק. כל המסננים הללו יחד מגבילים את הנתונים שניתן להזין בעותק חלקי.

מנהל השכפול

מנהל השכפול, הכלול במהדורת Office למפתחים יסייע בידך בניהול קבוצת עותקים משוכפלים ברשת, העשויה לכלול גם חיבורי LAN ו- WAN. במהדורה הראשונה שצורפה ל- Access 97, הכיל מנהל השכפול תמיכה בסינכרון דרך האינטרנט באמצעות FTP. אפשרות הסינכרון החדשה באמצעות האינטרנט הכלולה ב- Access 2000 הופכת תכונה זאת לבלתי נחוצה. עם זאת, עליך להמשיך ולהשתמש במנהל השכפול עם גרסת הסינכרון דרך האינטרנט הכלולה ב- Access 2000.

הפעל את מנהל השכפול כדי להתחבר באופן מזדמן אל רשת לצורך סינכרון עקיף, תהליך שבו השינויים מהעותק השולח עוברים לעותק המקבל דרך תיבה להשארת נתונים, אם העותק המקבל סגור באותה עת. במועד מאוחר יותר, כאשר מסד הנתונים המקבל ייפתח, סוכן הסינכרון שבשליטת מנהל השכפול יעביר אליו את העדכונים. מנהל השכפול כולל גם ממשק גרפי למשתמש לצורך תזמון של עדכונים במרווחי זמן קבועים. עליך להגדיר את תצורת מנהל השכפול עבור השרת שלך.

תוכל להשתמש במנהל השכפול גם לצורך תיאום בין עותקים ברחבי רשת תקשורת מרחבית (WAN). ניתן להציג בו באופן גרפי את הטופולוגיה של קבוצת העותקים המשוכפלים. אחד המבנים הנפוצים הוא טיפולוגיה של כוכב, שבה עותק מרכזי מחליף נתונים עם קבוצה משויכת של עותקי חישור (spoke replicas). כל אחד מעותקי החישור מתחבר רק אל העותק המרכזי. טופולוגיה של חיבורים מלאים מקשרת כל עותק במישרין עם כל שאר העותקים. דבר זה מאפשר העברה מהירה הרבה יותר של עדכוני נתונים ברחבי קבוצת העותקים המשוכפלים, אולם הדבר מתבטא בהגברת עומס התעבורה ברשת. טיפולוגיות מסוגים נוספים מציעות יחסי רווח-הפסד שונים בין מבנה לרמת ביצועים.

סינכרון דרך האינטרנט

בעזרת סינכרון דרך האינטרנט, תוכל לשכפל מסדי נתונים דרך חיבור FTP או HTTP 1.1 ב- Web. החיבור יכול להתבצע במסגרת האינטרנט, רשת ארגונית או רשת אקסטרא-נט ואין צורך בחיבור רציף בין המחשבים. שלא כמו הסינכרון הישיר

הנתמך במנהל השכפול, סינכרון דרך האינטרנט אינו מחייב הפעלה של סוכן סינכרון במחשב הלקוח. יתר על כן, סינכרון דרך האינטרט יכול לפעול עם עותקים אנונימיים. זוהי אחת מהגדרות המאפיין Visibility בעותק אשר שולבה לראשונה ב- Access 2000 (עיין במאמר שכותרתו Internet Synchronization with Microsoft Jet 4.0 (סינכרון דרך האינטרנט בעזרת Microsoft Jet 4.0) בכתובת http://support.microsoft.com, שם תמצא פרטים על התקנה, ניהול ובדיקה של מערכת לסינכרון דרך האינטרנט).

כפי שהזכרנו קודם, תכונת הסינכרון דרך האינטרנט הפכה את גרסת הסינכרון ב- Access 97 למיושנת. ארבעה שיפורים מרכזיים מייחדים את תכונת הסינכרון דרך האינטרנט ב- Access 2000 בהשוואה לגירסה הקודמת:

- תמיכה בשכפול דרך חיבור של פרוטוקול HTTP 1.1 (לרשותך תמיכה גם בשרתי Netscape באמצעות פרוטוקול זה).
- רמת ביצועים משופרת בהקשר עם הצפנה.
- תמיכה בעותק 'קל' חדש המותאם במיוחד לשכפול מבוסס-Web.
- מפתחות רישום חדשים לכיוונון עדין של פסקי הזמן של המסנכרן.

כאשר אתה בוחר שרת Web שעמו תבצע סינכרון, Access קובע באופן אוטומטי אם להשתמש בפרוטוקול HTTP או FTP. מלבד יצירת אפשרות להפעלה על גבי שרתי Netscape, פרוטוקול HTTP מאפשר לעותק לבצע סינכנון עם מסנכרן באינטרנט תחת המטריה של שרת proxy בעל תצורה מתאימה. Access 2000 אינו תומך במפורש בתצורה ההפוכה. אם העותק המקורי שלך אינו מוצפן, סינכרון דרך האינטרנט אינו מצפין באופן אוטומטי את העדכונים שהוא משגר אל עותק היעד לסינכרון. הגירסה הראשונית של שכפול דרך האינטרנט הצפינה באופן אוטומטי את כל העדכונים שנועדו להעברה דרך ה- Web. העותקים החדשים, האנונימיים והקלים, יכולים ליצור שכפול רק עם עותק האב שלהם בשרת ה- Web. את עותק האב שבשרת ה- Web, עליך לנהל בעזרת מנהל השכפול.

שינויים בעיצוב השכפול

כאשר אתה מאפשר שכפול של מסד נתונים באחד האמצעים המתוארים לעיל, בדרך כלל אתה מוסיף לכל טבלה אוסף של טבלאות מערכת, וכן קבוצה של שדות. הטבלאות והשדות הללו מסייעים בניהול פרויקט השכפול. עם זאת, הם עלולים להגדיל באופן ניכר את נפח מסד הנתונים. מעבר לכך, טבלאות ושדות השכפול המיוחדים גוזלים ממשאבי יישום מסד הנתונים, דבר שמפחית בו במקצת את מספר הטבלאות המותאמות אישית המירבי לכל מסד נתונים, את מספר השדות המותאמים אישית המירבי לכל טבלה, ואת מספר הבתים הזמינים לכל רשומה לשימושים מותאמים אישית ביישום מסד הנתונים. הבנה של שינויי העיצוב הללו תסייע בידך בניהול פרויקטים של שכפול ב- Access.



הערה:

כדי להציג את רוב טבלאות ושדות השכפול המיוחדים, בחר את הפריט **אפשרויות** (Options) מתוך תפריט **כלים** (Tools). בחר בתיבת הסימון **אובייקטים של המערכת** (System Objects), ולחץ על **אישור**.

שדות של מערכת שכפול

בדרך כלל מתווספים לטבלאות של יישום שכפול ארבעה שדות חדשים: s_GUID, s_Generation, s_Lineage ו- s_Collineage. השדה s_GUID מזהה באופן ייחודי כל שורה בכל טבלה בקבוצת העותקים המשוכפלים. בשני עותקים שונים, ייתנו לשורות זהות בטבלאות זהות ערכי GUID (מזהה גלובלי ייחודי) שונים. Access בונה את מחרוזות GUID ב-16 בתים באופן המעניק להן ייחוד גלובלי.

בתכנון טבלאות ליישום של שכפול, באפשרותך להפעיל מזהה GUID המופק באופן אוטומטי, כמפתח ראשי עבור מסד נתונים. אם מזהה GUID משמש כשדה מפתח ראשי של טבלה, Access אינו יוצר את השדה s_GUID כאשר אתה הופך טבלה מסוימת לניתנת לשכפול. במקום זה הוא משתמש במפתח הראשי לאותה מטרה. כדי להשתמש במזהה GUID כמפתח ראשי, בחר ב**קוד העתק משוכפל** (Replication ID) למאפיין של **גודל שדה** (FieldSize) עבור שדה בעל סוג הנתונים **מספר** (Number) או **מספור אוטומטי** (AutoNumber).

השדה s_Generation עוקב אחר הדור של שינוי שהוכנס בטבלה. הנתונים בשדה זה הם מסוג **מספר שלם ארוך** (Long Integer). הערך 0 מייצג שינוי חדש המחייב שכפול ללא תנאי. לאחר ההפצה הראשונית של שינוי אל עותק אחר, Access מעדכן את ערך השדה, כך שהוא מייצג את הדור החדש ביותר של שינויים. תהליך השכפול מנהל מעקב אחר הדור האחרון שנשלח לכל עותק מכל עותק אחר. כאשר נפתח תהליך חילופין חדש, Access ממשיך בדור החדש הבא של שינוי מאז הסינכרון האחרון שבוצע עם העותק.

השדה s_Lineage עוקב אחר היסטוריית השינויים שנעשו בכל שורה בטבלה. הנתונים בשדה זה הם מסוג אובייקט OLE. השדה מציין מתי שורה מסוימת נשלחת לעותק אחר. הוא מונע אפשרות של שיגור חוזר של שינוי שכבר נשלח לעותק אחר כלשהו.

השדה s_Collineage, המכיל נתונים מסוג **אובייקט OLE** (OLE Object), תומך בשכפול ברמת-עמודה. תכונה זאת חדשה ב- Access 2000. בגרסאות קודמות של Access נעשה שימוש בשכפול ברמת-שורה (בסעיף הבא, שכותרתו "חידושים בתחום השכפול ב-Office 2000" נעמיק את הדיון בנושא זה). השדה s_Collineage עוקב אחר מידע המאפשר לגלות שינויים ברמת-עמודה במהלך סינכרון.

כל שדה המכיל נתונים מסוג **תזכיר** (Memo) או מסוג **אובייקט OLE** מקבל גם שדה דור נפרד. מכיון ששדות אלה עשויים להיות גדולים במיוחד, אין הם מופצים, בהכרח, מעותק אחד למשנהו כאשר מתחלף ערך בשדה של שורה כלשהי. שדות אלה מופצים בין העותקים רק כאשר ערכם משתנה בפועל. שדות הדור הנפרדים שלהם עוקבים אחר

נתונים אלה לצורך תהליך השכפול. בגירסה הניתנת לשכפול של טבלת Category של Northwind יש שדה דור מיוחד אחד עבור הקטגוריה **Pictures**, אולם בטבלה Employees יש שני שדות דור מיוחדים – אחד עבור השדה Note, והשני עבור השדה Photo.

טבלאות של מערכות שכפול

מספר טבלאות מערכת תומכות בהתנהגות של קבוצת עותקים משוכפלים, כפי שמוצג בטבלה שלהלן. חלק מהטבלאות, כמו למשל MSysTableGuids, עשוי להיות מקוצר ובסיסי. בטבלה MSysTableGuids שמורים מזהי GUID עבור שמות כל הטבלאות בעותק נתון (להוציא טבלאות מערכת השכפול וטבלאות ההתנגשויות המיוחדות, המוסתרות). חלק מהטבלאות משתמשות במזהי GUID השמורים בטבלה זאת כדי לזהות טבלאות מוגדרות בקבוצת עותקים משוכפלים. בטבלאות אחרות נשמרים פרטים על פעולות היסטוריות או פעולות ממתינות. בטבלה MSysTombstone נשמרים פרטים אודות רשומות שנמחקו. הלוגיקה המוכללת לשכפול משתמשת בטבלה זאת כדי למחוק רשומות בעותקים המקבלים, במהלך הסינכרון.

טבלה	תיאור
MSysConflicts	ניהול מעקב אחר כל ההתנגשויות. טבלה זאת מופצת בין כל החברים בקבוצת עותקים משוכפלים.
MSysExchangeLog	ניהול מעקב אחר נתוני סינכרון בין עותק מסוים לבין כל שאר החברים בקבוצת עותקים משוכפלים.
MSysGenHistory	שמירת נתונים אודות כל דור של סינכרון. טבלה זאת מונעת משלוח של שינויים מדורות ישנים לחברים פעילים בקבוצת עותקים משוכפלים, וכן עדכון של עותקים ששוחזרו מתוך עותקי גיבוי.
MSysOthersHistory	שמירת נתונים אודות דורות של עדכונים מחברים אחרים בקבוצת עותקים משוכפלים.
MSysRepInfo	טבלה זאת מכילה רשומה אחת, ובה פרטים הרלוונטיים לקבוצת העותקים המשוכפלים כולה. טבלה זאת מופצת בין כל החברים בקבוצת עותקים משוכפלים.
MSysReplicas	בטבלה זאת שמורים נתונים על כל העותקים בקבוצת עותקים משוכפלים.
MSysSchChange	שמירת נתונים אודות שינויים בעותק המשמש תבנית הבסיס לעיצוב, לצורך הפצה בין שאר החברים בקבוצת העותקים המשוכפלים.
MSysSchemaProb	שמירת נתונים אודות התנגשויות בין החברים בקבוצת עותקים משוכפלים. אם אין התנגשויות בלתי פתורות, טבלה זאת לא תופיע.
MSysSchedule	טבלה זאת משמשת את סוכן הסינכרון המקומי לניהול התזמון של פעולות סינכרון עם עותקים אחרים.

טבלה	תיאור
MSysSideTable	טבלה זאת מכילה נתונים מפורטים על התנגשויות.
MSysTableGuids	טבלה זאת משייכת שמות טבלה למזהי GUID. גם מערכות שיכפול אחרות משתמשות במזהי GUID אלה.
MSysTombstone	שמירת נתונים היסטוריים אודות רשומות שנמחקו, ותמיכה בעדכוני מחיקה בכל רחבי קבוצת העותקים המשוכפלים.
MSysTranspAddress	שמירת נתונים אודות מסנכרנים המנהלים עותקים בקבוצת עותקים משוכפלים.
MSysContents	שמירת נתוני שורות עבור עותקים חלקיים. טבלה זאת תופיע רק בעותקים חלקיים.
MSysFilters	שמירת נתוני מסננים עבור עותקים חלקיים. טבלה זאת תופיע רק בעותקים חלקיים.

הטבלה MSysConflicts מפנה אל קבוצה של טבלאות שאמנם אינן טבלאות מערכת, אולם הן מוסתרות. בטבלאות אלה שמורים פרטים אודות התנגשויות ושגיאות נפרדות עבור כל טבלה. לפי כללי מתן השמות, שמה של טבלה מוסתרת כזאת יהיה UserTableName_Conflicts. לדוגמה, אם יש התנגשות אחת או יותר בין נתונים בשני עותקי הטבלה Employees, נוצרת עבור העותק ש'מפסיד' בהתנגשות טבלה בשם Employees_Conflict. השורות בטבלה מתעדות את הרשומה ש'הפסידה' ומכילות המלצה להמשך. כאשר משתמשים פותרים התנגשויות באמצעות לוגיקה מוכללת, Access תלוי בטבלאות אלה ומנהל אותן. אם תבנה לוגיקה מותאמת לפתרון התנגשויות, גם הפתרונות המותאמים שתבנה יצטרכו לנהל טבלאות אלה. לדוגמה, לאחר שנפתרו כל ההתנגשויות בטבלה מסוימת, על היישום שלך למחוק את הטבלה UserTableName_Conflict. פעולה זאת תמחק את השורה המקבילה מהטבלה MSysConflicts.

טבלאות ושדות השכפול מטילים מגבלות נוספות על העיצוב של יישומי Access. לדוגמה, Access מתיר הצבה של 255 שדות בטבלה, לכל היותר. בשכפולים, בדרך כלל נוספים בכל טבלה ארבעה שדות (s_GUID, s_Generation, s_Lineage ו- s_Collineage), ועוד שדה נוסף עבור כל שדה תזכיר או שדה אובייקט OLE. כאשר אתה מתכנן את השדות של טבלה, עליך להותיר מקום עבור שדות השכפול המיוחדים. שיקולים דומים חלים על מניין הבתים המירבי לכל רשומה. עיין במאמר שכותרתו Database Replication in Microsoft Jet 4.0 (שכפול מסדי נתונים ב- Microsoft Jet 4.0) בכתובת http://support.microsoft.com, שם תמצא הנחיות בנושאים אלה ובנושאים אחרים בתחום העיצוב המתקדם של שכפולים.

גיבוי ושחזור מסדי הנתונים המקוריים

עליך לשקול ביסודיות אם רצונך להפוך קובץ כלשהו לבר-שכפול, מכיון שלא קיימת תכונה מוכללת לשחזור מסד הנתונים המקורי. מסיבה זאת, אולי רצוי שתיצור גיבוי של קובץ מסד הנתונים בטרם תאפשר לשכפל אותו.

אם ידועים לך השמות של טבלאות ושדות השכפול, תוכל לשחזר מתוך אחד העותקים של קבוצת עותקים משוכפלים קובץ מסד נתונים חדש שאינו בר-שכפול ואשר יכיל את הנתונים הנוכחיים של אותו עותק. עליך לצרף טבלאות מהעותק הנבחר אל העותק החדש של מסד הנתונים. צרף רק את שדות המשתמש, ולא את שדות השכפול המיוחדים. תוכל לייבא את כל הטבלאות במקום זאת (להוציא, כמובן, טבלאות השכפול המיוחדות), ולבצע שאילתות מחיקה כדי להסיר את שדות השכפול המיוחדים. הוסף את קשרי הגומלין בין הטבלאות, ולאחר מכן ייצא את שאר אובייקטי מסד הנתונים.

חידושים בתחום השכפול ב- Access 2000

אחד מגורמי השדרוג החשובים ביותר ב- Access 2000 הוא השקת מבנה JRO לשליטה בשכפול באמצעות קוד. הואיל וחלק מהתכונות החדשות, כגון המאפיין Visibility, אינו זמין בממשק תכנות DAO המקובל – גם לא בגירסה 3.6 העדכנית ביותר – אין ספק שיהיה עליך ללמוד את השיטה החדשה לתכנות שכפולים.

שכפול דו-כיווני ב- Jet-SQL Server

Access 2000 מאפשר שכפול דו-כיווני בין עותקי Jet ו- Microsoft SQL Server. העברה מסוג זה מחייבת עבודה עם Jet 4 ו- SQL Server 7. העברה דו-כיוונית פירושה ש-SQL Server יכול לשמש כמחסן מרכזי עבור קבוצת יישומי Access שאינם מחוברים. עובדים שעבודתם מחייבת ניידות, והמפעילים יישומי Access, יוכלו להעביר עדכונים למסד נתונים מרכזי, ולטעון את השינויים העדכניים ביותר מתוך מסד הנתונים המרכזי. עליך להתחיל בעותק משוכפל של SQL Server, או להגדיל עותק משוכפל ב-Access למימדי SQL Server.

כאשר אתה מגדיר קבוצת עותקים משוכפלים של Jet ו- SQL Server, חובה להציב עותק של SQL Server במרכז, ואילו עותקי Jet יוכלו לפעול כחישורים. עותקי Jet יוכלו לבצע חילופי תכנים דו-כיווניים עם עותק SQL Server מרכזי, אולם לא יוכלו לבצע זאת בינם לבין עצמם. על עותק SQL Server המרכזי לשמש תמיד כמתווך בין כל זוג נתון של עותקי Jet המוצבים סביבו. הואיל ו- SQL Server אינו אלא מנגנון של מסד נתונים, שאינו מכיל סביבת פיתוח יישומים מלאה כמו Access, לא תוכל לשכפל אובייקטי יישומים ייחודיים של Access, כמו טפסים ודוחות, אל תוך SQL Server מרכזי. עם זאת, עיצוב מסוג זה מפצה על חיסרון הנזכר לעיל בכך שהוא מאפשר שימוש בשאר היתרונות המיוחדים של מערכות SQL Server, כמו למשל עיבוד שרת-לקוח, ואפשרויות דירוג של ריבוי מעבדים (עיין בנושא Implementing Merge Replication to Access Subscribers ב- SQL Server Books Online, שם תמצא פרטים נוספים על ההתנהגות של קבוצות עותקים משוכפלים משותפות ל- Jet ו- SQL Server).

עדכונים ברמת-עמודה

אחת השיטות החשובות להגברת התפוקה בקרב עובדים המשתמשים במסד נתונים בר-שכפול היא לצמצם את מספר ההתנגשויות. Access 2000 מאפשר לראשונה עדכונים ברמת-עמודה במטרה למזער את מספר ההתנגשויות בין שני עותקים נתונים. בגרסאות קודמות של Access, עדכונים משני עותקים שונים התנגשו, אם נערך בהם שינוי באותה רשומה – גם אם שינו שני שדות שונים באותה רשומה – מכיון שרמת המעקב הנמוכה ביותר אחר עדכונים היתה רמת השורה. ב- Access 2000 תוכל ליצור עותקים המזהים שינויים עד לרמת השדות הנפרדים. לכן, משתמש אחד יכול לשנות את מספר הפקס של לקוח בעותק אחד, בעוד שהמשתמש השני משנה את מענו של איש הקשר עבור אותו לקוח, בעותק אחר. כאשר נערך סינכרון בין שני העותקים, לא מתרחשת התנגשות.

מעקב ברמת-עמודה הוא ברירת המחדל עבור כל העותקים החדשים. אם תעדיף זאת, תוכל לבחור במעקב המקובל, ברמת-שורה. כאשר מעדכנים מסד נתונים בר-שכפול מתוך גירסה קודמת של Access, הוא משמר את הגדרת המעקב ברמת-שורה. מכיון שמחירו של מעקב ברמת-עמודה גבוה במונחים של רמת הביצועים והגודל, רצוי שלא להשתמש בו אם הסבירות להתנגשויות נמוכה. התכונה מעקב ברמת-עמודה פועלת גם עם עותקי SQL Server.

רמות החשיפה של עותק משוכפל

Access 2000 מכיל לראשונה שלוש רמות חשיפה של עותקים משוכפלים: גלובלי, מקומי ואנונימי. באפשרותך לשלוט ברמת החשיפה של עותק משוכפל כמאפיין ב- JRO. לא ניתן לשנות את רמת החשיפה של עותק משוכפל לאחר שנוצר.

עותק משוכפל עם רמת חשיפה גלובלית זהה בתיפקודו לעותקים המשוכפלים המקובלים ב- Access. הם יכולים לבצע שכפול מול כל עותק משוכפל אחר, ונראים לעין בכל רחבי קבוצת העותקים המשוכפלים. לעותקים משוכפלים עם רמת חשיפה מקומית או אנונימית תפקידים מיוחדים המאפשרים צמצום בגודלם בהשוואה לעותקים הגלובליים הרגילים. עותקים משוכפלים שנוצרו מעותקים מקומיים ואנונימיים יורשים את הגדרת מאפיין החשיפה של עותק האב, אולם יש להם מאפיין ReplicaId ייחודי. לא ניתן ליצור תבנית בסיס לעיצוב מעותק משוכפל מקומי או אנונימי.

עותקים מקומיים חשופים רק לעין עותק האב, והם עורכים חילופי תכנים אך ורק עם עותק האב. היוזמה לחילופי תכנים יכולה להיות הן של עותק האב והן של העותק המקומי. נוסף לכך, עותק האב המרכזי של עותק מקומי יכול לתזמן אירועי סינכרון חוזרים מול עותק מקומי. עותקים מקומיים אינם יכולים לערוך חילופי נתונים במישרין עם עותקים אחרים בקבוצת עותקים משוכפלים. עם זאת, ניתן להפיץ שינויים בעותק מקומי, דרך עותק האב. התנגשויות בין עותק מקומי לעותק האב מסתיימים תמיד בנצחון האב.

עותקים אנונימיים מיועדים להפצה דרך חיבור מבוסס-Web (FTP או HTTP 1.1). כדי לבצע בהצלחה סינכרון בין עותק לקוח אנונימי לבין עותק מרכזי ב- Web, המקור של העותק האנונימי המקורי צריך להיות עותק גלובלי המנוהל באמצעות מנהל השכפול בשרת ה- Web. תוכל להפיץ עותקים בכל אמצעי מתאים (למשל, דרך ה- Web או באמצעות תקליטור). בדומה לעותק המקומי, העותק האנונימי יכול לערוך סינכרון רק עם עותק האב, אולם האב אינו יכול לתזמן שכפולים עם עותקי הצאצא האנונימיים שלו. חילופי הנתונים נערכים תמיד ביוזמת העותק האנונימי. התנגשויות במהלך הסינכרון מסתיימות תמיד בנצחון עותק האב.

פתרון התנגשויות המבוסס על קדימויות

ב- Access 2000 חל כלל חדש כברירת מחדל לפתרון התנגשויות. במהדורות הקודמות של Access ההכרעה בהתנגשויות בין עותקים נפלה לטובת העותק שביצע את מספר השינויים הגדול ביותר ברשומה נתונה. אם שני עותקים משוכפלים ביצעו מספר זהה של שינויים ברשומה, ניצח העותק בעל הערך הנמוך ביותר במאפיין ReplicaId. בערכה החדשה לפתרון התנגשויות, ברירת המחדל מחילה את הכלל: העותק בעל הקדימות הגבוהה יותר הוא זה שינצח. הערכים בהגדרות מאפיין הקדימות של עותקים עשויים לנוע בין 0 ל-100. אם לשני עותקים קדימות זהה, העותק בעל הערך הנמוך ביותר במאפיין ReplicaId הוא זה שינצח. יתרונו של הכלל החדש בכך שהוא עולה בקנה אחד עם הכלל החל בשכפול ב- SQL Server 7.

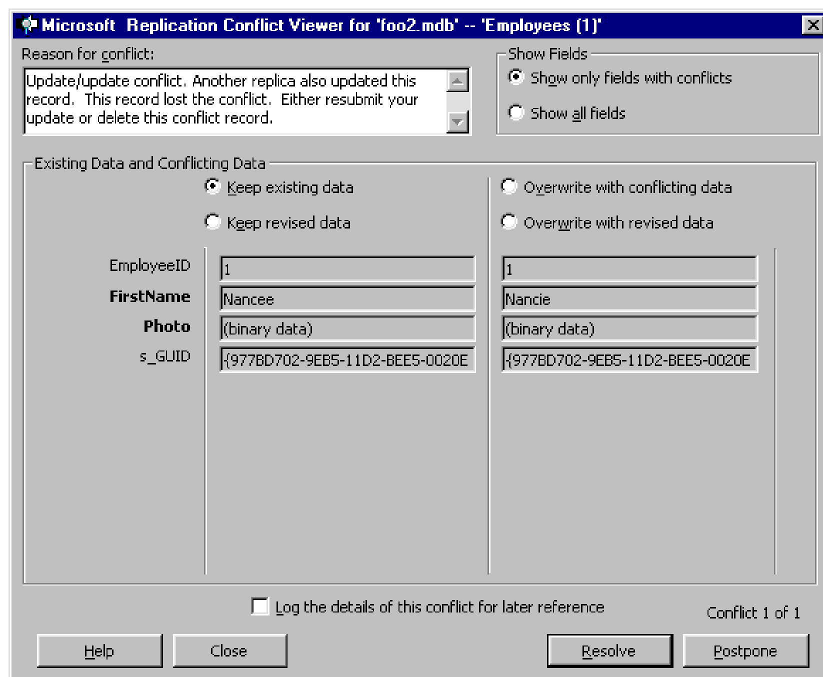
הגדרת המאפיין Priority (קדימות) בעותק משוכפל ניתנת לקריאה בלבד מרגע שהעותק נוצר. לפי ברירת המחדל, ניתנת לעותק הראשוני של מסד נתונים הגדרת קדימות של 90. הקדימות שמקבל עותק גלובלי המבוסס על עותק אחר היא 90 אחוזים מן הקדימות של העותק הראשוני. בעותקים אנונימיים ומקומיים נכפה ערך 0 במאפיין Priority. עותקים משוכפלים שהועתקו באמצעות MS-DOS או באמצעות השיטה CompactDatabase מקבלים ערך קדימות התואם את המקור. מסדי נתונים שעברו המרה מקבלים ערך קדימות של 90. בכל מקרה אחר, תוכל להציב במאפיין Priority כל ערך בטווח החוקי בעת שאתה מאפשר שכפול של מסד נתונים. חובה להציב בהגדרת המאפיין Priority של כל העותקים הגלובליים שייווצרו בהמשך ערך שווה לזה של עותק האב, או נמוך ממנו (אלא אם כן האדם שיוצר את העותק המשוכפל חבר בקבוצה Admins, או שהינו הבעלים של מסד הנתונים).

שכלולים שונים

סדרה של התאמות שונות משלימה את פונקציונליות השכפול החדשה ב-Access 2000. שתיים מהן מפשטות משימות שגרתיות כמו הגבלת ההתנהגות של עותק משוכפל, או פתרון שגיאות סינכרון. במקרה אחר, שינוי בעיצוב עשוי להשפיע על השיטה המיטבית להפצת שינויים בעיצוב היישום. התכונה החדשה **מנע מחיקות** (Prevent Deletes) שתוארה קודם אינה מאפשרת למשתמש למחוק רשומות. באמצעות תכונה זאת ניתן בנקל למנוע ממשתמשים בלתי מנוסים מחיקה בשוגג של תכנים חשובים. תכונה זאת

אינה מונעת הפצה של מחיקות אל עותק זה מעותק אחר, למשל עותק המנוהל בידי מנהל מסד נתונים.

כפי שכבר הזכרנו, ניתן כעת לטפל בהתנגשויות ובשגיאות סינכרון באותו ממשק. האשף פתרון התנגשויות (Conflict Resolution) מציג הן התנגשויות והן שגיאות (ראה תרשים 11.2). דבר זה מבטל את הצורך בעיבוד נפרד באמצעות שני ממשקים. נוסף לכך, האשף פועל הן בשכפולי Jet 4 והן בשכפולי SQL Server 7.



תרשים 11.2: Microsoft Conflict Viewer של אשף פתרון התנגשויות מראה הן התנגשויות והן שגיאות

בשעת סינכרון, מחיקות מקבלות תמיד קדימות גבוהה יותר מכל שינוי אחר הקשור ברשומה. דבר זה נכון בכל גרסאות Access, אולם ב- Access 2000, רשומה ש׳הפסידה׳ מול פעולת מחיקה, מוסיפה ערך בטבלת ההתנגשויות של טבלת המשתמש. הגרסאות הקודמות התעלמו מעדכונים ש׳הפסידו׳ מול פעולת מחיקה.

Access 2000 כולל לראשונה תמיכה בהיררכיה של רשומות מתנגשות. אם יש התנגשות בין המפתחות הראשיים בשני עותקים, האחד ׳ינצח׳ והשני ׳יפסיד׳. הרשומות בטבלאות האחרות הקשורות לאותה טבלה יפסידו אף הן. כאשר אתה מתקן את המפתח הראשי בעותק המפסיד, התיקון ב- Access 2000 עובר בהיררכית הרשומות הקשורות לטבלה זאת בטבלאות אחרות, כך שאין צורך בהתאמות נפרדות. בגרסאות קודמות של Access נדרשו תיקונים נפרדים בטבלה הראשית ובטבלאות הקשורות אליה.

Access 2000 מכיל תבנית אחסון חדשה העשויה להשפיע על החלטתך כיצד להפיץ שינויי תוכנה בפרויקט משוכפל. בגרסאות קודמות ניתן היה לסנכרן שינויים באובייקטים נפרדים של Access, כמו למשל טפסים ודוחות. ב-Access 2000, אובייקטים של Access כמו טפסים, דוחות, מודולים ועמודים נמצאים באובייקט בינארי גדול יחיד, או בקובץ "project.adp" הנפרד. שיטת האחסון החדשה כופה שכפול של כל האובייקטים אם מתעורר צורך לעדכן אובייקט מסוים. אם פתרון זה אינו אטרקטיבי, תוכל להפוך את פרויקט Access שבתבנית הבסיס לעיצוב לבלתי ניתן לשכפול. במקרה זה תוכל להפיץ ללא הגבלה שינויי עיצוב, באמצעים אחרים (למשל, תקליטור).

טכניקות פיתוח ב- JRO

סעיף זה מציג טכניקות פיתוח ב-JRO לעבודה עם שכפולים, ובין היתר הפיכת מסד נתונים לבר-שכפול, יצירת עותק משוכפל מלא או חלקי, דחיסת מסדי נתונים, סינכרון מסדי נתונים, ותיעוד מאפיינים של עותק משוכפל.

הפיכת מסד נתונים לבר-שכפול

השתמש בשיטה MakeReplicable של אובייקט Replica כדי להפוך מסד נתונים לבר-שכפול. תהליך זה גורם לעותק המשוכפל החדש להפוך לתבנית בסיס לעיצוב, עם הגדרת חשיפה גלובלית במאפיין Visibility. הפיכת מסד נתונים רגיל של Access למסד נתונים בר-שכפול מוסיפה את השדות והטבלאות המיוחדים אשר תוארו קודם. שדות וטבלאות אלה עשויים להגדיל באופן ניכר את נפח מסד הנתונים, לכן רצוי להכין עותק גיבוי של מסד הנתונים בטרם תהפוך אותו לבר-שכפול.

להלן תחביר השיטה MakeReplicable:

```
Replica.MakeReplicable ConnectionString, ColumnTracking
```

מחרוזת החיבור מצביעה אל מסד הנתונים שברצונך להמיר לתבנית ניתנת לשכפול. באפשרותך להגדיר את המאפיין ActiveConnection של האובייקט Replica בטרם תפעיל את השיטה MakeReplicable. עם זאת, ארגומנט מחרוזת החיבור של השיטה עוקף את המאפיין ActiveConnection של העותק המשוכפל. הארגומנט ColumnTracking של השיטה הוא משתנה בוליאני. ערך ברירת המחדל שלו הוא True. זכור, שדבר זה עשוי, להלכה, לסייע בצמצום ההתנגשויות בסינכרון. אם אין זה סביר שיתרחשו התנגשויות (לדוגמה, מכיון שכל העריכה תתבצע בעותק משוכפל יחיד), עליך לשקול הצבת ערך False במשתנה. במקרה כזה, המעקב אחר התנגשויות בסינכרון יבוצע בשיטה הרגילה של מעקב ברמת-שורה. דבר זה מונע את הנפילה ברמת הביצועים הצפויה במקרה של מעקב ברמת-עמודה.

השגרות הבאות מחילות את השיטה MakeReplicable על מסד הנתונים של Northwind בשעה שמסד הנתונים מבצע פונקציות של גיבוי ולכידת שגיאות. השיגרה callMakeDesignMaster מציבה ערכים בארגומנטים שלה path ו-replicaName, ולאחר מכן היא מפעילה את השיגרה makeDesignMaster. בדוגמה מבוצע שרשור של משתני path ו- replicaName תוך הפעלת השיגרה השנייה. השיגרה השנייה מפעילה את השיטה MakeReplicable עבור הערכים שהועברו אליה.

```
Sub callMakeDesignMaster()
   path = "C:\Program Files\Microsoft Office\Office\Samples\"
   replicaName = "Northwind.mdb"
' Set the second parameter to True to invoke
' column-level tracking of resolution conflicts.
   makeDesignMaster path & replicaName, True
End Sub

Sub makeDesignMaster(newReplica As String, _
   Optional ColumnTracking As Boolean)
On Error GoTo DMTrap
Dim repMaster As New JRO.Replica
' Offer to copy database for restoring it after making the database replicable.
   If MsgBox("Do you want to make a backup copy", _
      vbYesNo, "Programming Microsoft Access 2000") = vbYes Then
      Set fs = CreateObject("Scripting.FileSystemObject")
      fs.Copyfile newReplica, "c:\My Documents\DMBackup.mdb"
   End If

' Optionally make the newReplica database replicable.
   If ColumnTracking = True Then
      repMaster.MakeReplicable newReplica
   Else
      repMaster.MakeReplicable newReplica, False
   End If
' Clear reference to Design master.
   Set repMaster = Nothing
DMExit:
   Exit Sub

DMTrap:
   If Err.Number = -2147467259 And _
      Left(Err.Description, 5) = "Could" Then
      MsgBox "Can not create replica because file does " & _
         "not exist. Fix path/file name and try again.", _
         vbCritical, "Programming Microsoft Access 2000"
      Resume DMExit
```

```
    ElseIf Err.Number = -2147467259 And _
        Left(Err.Description, 8) = "Database" Then
        MsgBox "Database is already replicable. Use the " & _
        "CreateReplica method to base a new replica " & _
        "on it.", vbCritical, "Programming Microsoft Access 2000"
        Resume DMExit
    ElseIf Err.Number = 53 Then
        MsgBox "Original file not found for backup copy. " & _
            "Correct file name and try again.", vbCritical, _
            "Programming Microsoft Access 2000"
        Resume DMExit
    Else
        Debug.Print Err.Number; Err.Description
    End If

End Sub
```

לפני הפעלת השיטה MakeReplicable, השיגרה makeDesignMaster שואלת את המשתמש אם ברצונו ליצור עותק גיבוי. אם המשתמש משיב **כן**, השיגרה יוצרת מופע של האובייקט FileSystemObject, ולאחר מכן מפעילה את השיטה Copyfile כדי ליצור גיבוי לקובץ. דבר זה מאפשר בנקל לחזור לגרסת המקור של מסד הנתונים, ללא שדות וטבלאות השכפול המיוחדים.

השיגרה makeDesignMaster מקבלת ארגומנט אחד, או שניים. הראשון הוא השרשור של path ו- replicaName. תבנית הבסיס לעיצוב של קבוצת העותקים המשוכפלים החדשה היא קובץ ששמו נקבע לפי ערך הארגומנט replicaName. השיגרה השנייה יכולה לקבל גם ארגומנט נוסף. אם אפשרות זאת מתממשת, מדובר במשתנה בוליאני המציין אם יש להפעיל מעקב ברמת-עמודה אחר התנגשויות בסינכרון. דוגמה זאת מעבירה ערך True. עיצוב הדוגמה מחייב את המשתמש לציין True בארגומנט השני כדי לממש מעקב ברמת-עמודה. אי-ציון הארגומנט השני יגרום למשתנה הבוליאני להניח כי ערך ברירת המחדל שלו, False. הואיל והשיטה MakeReplicable יוצרת לפי ברירת המחדל עותקים משוכפלים עם מעקב ברמת-עמודה, למעשה השיגרה אינה חייבת לציין בפועל True כדי ליצור עותק משוכפל עם תכונה זאת.



הערה:

אין בנמצא מאפיין למעקב ברמת-עמודה. לכן, עליך לעקוב ידנית אחר המצב של משתנה זה בכל העותקים המשוכפלים שלך.

השיגרה לוכדת במפורש שלוש שגיאות שונות. האחת היא בהפעלת האובייקט FileSystemObject, ושתי האחרות מקורן ב- Jet. שים לב, שמרכיב השכפול ב- Jet מעביר בחזרה מספר Err זהה (-2147467259) עבור שתי שגיאות שונות במובהק. למרבה המזל, לשגיאות אלה תיאורים שונים. הדוגמה בעמוד הקודם משתמשת

בתכונה זאת כדי להבדיל בין השתיים (במערכת עובדת יש להשתמש במקטע תיאור ארוך יותר כדי לזהות בוודאות את סוג השגיאה, או להשתמש בטכניקה מתקדמת יותר לניתוח שגיאות).



הערה:

ניתן להבדיל בין שגיאה מקורית לשגיאות Jet מבלי להסתמך על המאפיין ErrDescription. עם זאת, למאפיין זה נודעת חשיבות פוטנציאלית עבור מפתחים רבים. הטכניקה השנייה כרוכה בספירת האוסף Errors באובייקט Connection. פעולה זאת מחזירה ממנגנון מסד הנתונים המקורי מספרי שגיאה מובחנים.

יצירת עותקים מלאים נוספים

החל את השיטה CreateReplica על מופע חדש של אובייקט Replica כדי להפוך את המופע לחבר חדש בקבוצת עותקים משוכפלים. בטרם תפעיל את השיטה, הגדר עבור המופע החדש מאפיין ActiveConnection, כך שיצביע אל תבנית בסיס לעיצוב, או אל עותק משוכפל אחר מתוך קבוצת המטרה של עותקים משוכפלים. שיטה זאת תיכשל אם ההגדרה ActiveConnection מציינת בשגגה מסד נתונים אשר במאפיין ReplicaType שלו הוצב ערך jrRepTypeNotReplicable. ככלל, שיטה זאת מחזירה עותק משוכפל בעל סוג וחשיפה זהים למקור. עם זאת, מכיון שאמורה להתקיים רק תבנית בסיס אחת לעיצוב, יצירת עותק משוכפל לפי הדגם של תבנית הבסיס לעיצוב מחזירה עותק משוכפל גלובלי נוסף, שגם הוא כפוף לפרמטרים עבור השיטה. התחביר הכללי להחלת השיטה מובא להלן:

```
Replica.CreateReplica ReplicaName, Description, ReplicaType, _
   Visibility, Priority, Updatability
```

הפרמטר ReplicaName מציין את הנתיב ואת שם הקובץ של העותק המשוכפל החדש, אמצעות 255 תווים לכל היותר. Description הוא שדה אופציונלי המסייע בזיהוי חברים בקבוצת עותקים משוכפלים. ערך ברירת המחדל לפרמטר ReplicaName הוא jrRepTypeFull, ליצירת עותק מלא. תוכל לציין במקומו את הערך jrRepTypePartial. הפרמטר Visibility יכול לקבל את אחד הערכים הבאים: jrRepVisibilityGlobal (גלובלי, ברירת המחדל), jrRepVisibilityLocal (מקומי) או jrRepVisibilityAnon (אנונימי). אם לא תציין ערך עבור הפרמטר Priority, יינקטו כללי ברירת המחדל; הטווח המירבי הוא מ-0 ועד 100, כולל. לפי ברירת המחדל, עותק משוכפל מלא מקבל 90 אחוזים מערך הפרמטר Priority של עותק האב. הפרמטר Updatability קובע אם העותק החדש יהיה לקריאה בלבד (הערך jrRepUpdReadOnly), או לקריאה וכתיבה כאחת (הערך jrRepUPdFull).

הדוגמה הבאה יוצרת עותק משוכפל המבוסס על תבנית הבסיס לעיצוב של Northwind מן הדוגמה הקודמת. ההגדרה ActiveConnection יוצרת את העותק המשוכפל. העותק המשוכפל החדש הוא עותק מלא, עם רמת חשיפה גלובלית. הנתיב אל העותק המשוכפל הוא C:\My Documents\foo.mdb. בהתאם להגדרות הפרמטרים של העותק המשוכפל החדש, תיאורו הוא "foo full replica".

```
Sub makeFullReplica()
Dim repMaster As New JRO.Replica
' Point repMaster at a design master mdb.
    repMaster.ActiveConnection = _
    "C:\Program Files\Microsoft Office\Office\Samples\Northwind.mdb"

' Make sure foo.mdb is deleted before running the next line.
    repMaster.CreateReplica "c:\My Documents\foo.mdb", "foo full replica", _
         jrRepTypeFull, jrRepVisibilityGlobal, , jrRepUpdFull
End Sub
```

אין צורך ליצור גיבוי במקרה זה, הואיל והפקודה יוצרת עותק משוכפל חדש שחייב להתבסס על עותק קיים. עם זאת, השיגרה עלולה להיכשל אם העותק המשוכפל כבר קיים, או אם התבנית האמורה לשמש ליצירת העותק אינה קיימת (כלומר, הקובץ חסר), אולם מדובר בסוגיות פשוטות של לכידת שגיאות. הדוגמה הראשונית נקטה גישה מסוימת למשימה זאת, תוך יצירת עותק משוכפל. ייתכן שתרצה להניח, שהגירסה החדשה הופכת כל גירסה קיימת בעלת שם זהה למיושנת. הדוגמה השנייה ממשת את הלוגיקה הזאת בכך שהיא מוחקת את העותק המשוכפל הישן, פעולה המבטלת את הגורם האפשרי לאחת השגיאות עוד לפני שהשגיאה צצה.

יצירת עותק משוכפל חלקי, ומסננים

עותק משוכפל חלקי הוא עותק משוכפל שאינו מכיל את כל הנתונים שאמור להכיל עותק מלא. עותק משוכפל מסוג זה שימושי בעבודת סניפים או עובדים ניידים, הזקוקים לגישה אל קבוצת משנה מתוך הנתונים המוחזקים במטה. עותק חלקי מאפשר למשתמש לעיין רק בחלק מהנתונים המלאים, והדבר מפחית את היקף העדכון שנדרש כדי לסנכרן את העותק המשוכפל.

לאחר שיצרת עותק משוכפל חלקי וריק בעזרת השיטה CreateReplica, עליך לאכלס אותו בנתונים. לכל עותק חלקי אוסף Filters (מסננים). במסגרת אוסף זה, כל אובייקט Filter מציין נתח נתונים שונה שייכלל בעותק המשוכפל החלקי. כדי שעותק משוכפל חלקי יכיל נתונים כלשהם, עליך לציין עבורו מסנן אחד או יותר, לצרף את המסנן או המסננים אל אוסף Filters של העותק המשוכפל, ולאחר מכן להפעיל את השיטה PopulatePartial.

אפשר לבסס מסנן על הפסוקית WHERE מתוך משפט SQL (ללא מילת המפתח WHERE), או על יחס. הוסף וציין מסננים עבור עותק משוכפל חלקי בעזרת השיטה Append של האוסף Filters. שיטה זאת מקבלת שלושה ארגומנטים: המאפיין TableName קובע את הטבלה שהמסנן יחול עליה, המאפיין FilterType קובע בעזרת קבוע מי יסנן את הערכים עבור הטבלה, משפט קריטריונים ב-SQL (הערך jrFilterTypeTable), או יחס (הערך jrFilterTypeRelationship). המאפיין FilterCriteria כולל את שם היחס או את הפסוקית WHERE מתוך משפט SQL המגביל את הרשומות של טבלה.

השיטה PopulatePartial עבור עותק משוכפל מנקה את כל הרשומות בעותק המשוכפל החלקי ומאכלסת את העותק בהתאם למסננים החלים עליו. היא עושה זאת באמצעות סינכרון של העותק החלקי עם העותק המלא. השיטה מקבלת שני ארגומנטים. הראשון הוא משתנה אובייקט המצביע אל העותק החלקי שיש לאכלס מחדש. השני הוא משתנה מחרוזת, המגדיר את הנתיב ואת שם הקובץ של העותק המלא, שמולו מתבצע הסינכרון של העותק המשוכפל החלקי. ככלל, עליך להשתמש בשיטה PopulatePartial עבור עותק חלקי כאשר אתה יוצר את העותק החלקי, או אם שינית את המסננים שלו. כדי להפעיל את השיטה עבור עותק משוכפל חלקי, עליך לפתוח תחילה את העותק הרצוי עם גישה בלעדית, הואיל ובשלב הראשון, השיטה מנקה את כל הרשומות מן העותק החלקי כדי לאכלסו מחדש ברשומות.

ארבע השגרות הבאות מדגימות את אחת הגישות להגדרת שני עותקים חלקיים באמצעות האוסף Filters והשיטה PopulatePartial. השיגרה callMakePartialFilter מפעילה את השיגרה makePartialFilter פעמיים, כל פעם עם קבוצת ארגומנטים שונה. תחילה היא מפעילה את השיגרה כדי ליצור עותק משוכפל חלקי ששמו "Partial of Northwind.mdb". לאחר מכן היא חוזרת על התהליך ויוצרת עותק משוכפל נוסף בשם "Partial of foo.mdb".

```
Sub callMakePartialFilter()
   makePartialFilter "Partial of Northwind.mdb", _
      "Northwind.mdb", "C:\Program Files\Microsoft Office\Office\Samples\"
   makePartialFilter "Partial of foo.mdb", "foo.mdb", "C:\My Documents\"
End Sub

Sub makePartialFilter(replicaName As String, _
   sourceName As String, path As String)
Dim rep As New JRO.Replica
Dim flt1 As JRO.Filter
' Delete old partial.
   strfile = path & replicaName
   deleteFile (strfile)

' Make partial.
   makePartial path, replicaName, sourceName
```

```
' Open connection to partial and append filter.
    rep.ActiveConnection = _
        "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & _
        path & replicaName & ";Mode=Share Exclusive"
    rep.Filters.Append "Employees", jrFilterTypeTable, "Title='Sales Representative'"
    rep.Filters.Append "Customers", jrFilterTypeTable, _
        "Country='Spain' AND City='Madrid'"

' Populate partial from source.
    rep.PopulatePartial path & sourceName
End Sub

Sub deleteFile(strfile)
On Error GoTo deleteTrap
Dim cnn1 As New ADODB.Connection
' Prepare to delete file.
    Set fs = CreateObject("Scripting.FileSystemObject")
    fs.deleteFile strfile

deleteExit:
    Exit Sub
deleteTrap:
    If Err.Number = 70 Or Err.Number = 75 Then
        MsgBox "Partial is unavailable to system. " & _
            "Close it so that the system can create a " & _
            "new one.", vbCritical, "Programming Microsoft Access 2000"
    ElseIf Err.Number = 53 Then
        Resume Next
    Else
        Debug.Print Err.Number; Err.Description
    End If
    Resume deleteExit
End Sub

Sub makePartial(path As String, replicaName As String, sourceName As String)
Dim rep As New JRO.Replica

    rep.ActiveConnection = path & sourceName
    rep.CreateReplica path & replicaName, replicaName, jrRepTypePartial, _
        jrRepVisibilityGlobal, , jrRepUpdFull
End Sub
```

השיגרה makePartialFilter מקבלת ארגומנטים מהשיגרה callMakePartialFilter. ארגומנטים אלה מציינים את שם העותק המשוכפל החלקי שייווצר, ואת העותק המלא שהוא מקור הנתונים עבור העותק החלקי. השיגרה makePartialFilter גם מגדירה ומצרפת את אובייקטי Filter עבור העותק החלקי, ומפעילה את השיטה PopulatePartial כדי להחיל את המסננים.

תחילה, השיגרה מוחקת את כל העותקים החלקיים הקיימים, ששמם והמיקום שלהם זהים לאלו של העותק החלקי שהיא אמורה ליצור. פעולה זאת נעשית על ידי הגדרת משתנה מחרוזת המבוסס על המשתנים path ו- replicaName ששיגרה זאת מקבלת מהשיגרה הקוראת. לאחר מכן משתנה המחרוזת החדש מועבר אל השיגרה deleteFile. אלא אם מתרחשת שגיאה בלתי צפויה, השיגרה deleteFile מבצעת אחת משלוש משימות: היא מוחקת את הקובץ הישן של העותק המשוכפל, או שהיא מזכירה למשתמש לסגור עותק משוכפל פתוח כדי שהיישום יוכל למחוק אותו, או שהיא מתעלמת משגיאה שנגרמת מכך שהקובץ אינו קיים (קוד שגיאה 53).

לאחר שהשיגרה מוחקת, או מנסה למחוק, את הקובץ הקיים, היא יוצרת עותק חלקי חדש על ידי הפעלת השיגרה makePartial. הקריאה מעבירה שלושה ארגומנטים: path, replicaName ו- sourceName. השיגרה makePartial זהה כמעט לגמרי בעיצובה לשיגרה makeFullReplica. מלבד השימוש במשתנים לציון הנתיב ושם הקובץ, ההבדל המרכזי הוא שהשיגרה makePartial מציבה ערך jrRepTypePartial במאפיין ReplicaType, בעוד שהשיגרה makeFullReplica מציבה במאפיין זה את הערך jrRepTypeFull. שים לב, שהשיגרה makePartial מציינת את שם העותק החדש כשרשור של המשתנים path ו-replicaName. המאפיין ActiveConnection של מופע העותק החדש מציין את העותק המלא המשמש מקור לעותק החלקי. השיגרה מציינת מקור זה כשרשור של המשתנים path ו- sourceName. לשם כך חובה לשמור את העותק המלא ואת העותק החלקי באותו נתיב, אולם אין קושי בהסרת מגבלה זאת. הדוגמה האחרונה בפרק זה מראה כיצד לעשות זאת.

לאחר שהשיגרה makePartial מחזירה את השליטה לשיגרה makePartialFilter, העותק החלקי החדש קיים, אולם אינו מכיל נתונים. השיגרה makePartialFilter מאכלסת אותו בנתונים. תחילה היא מציבה את המאפיין ActiveConnection של מופע העותק בעותק החלקי החדש, ופותחת את העותק החדש במצב של 'גישה בלעדית'. זכור, שדבר זה נדרש כדי להפעיל את השיטה PopulatePartial. בשלב הבא השיגרה מגדירה ומצרפת לעותק שני אובייקטי Filter. המסנן הראשון מחלץ נתוני 'עובדים' שתפקידם מוגדר כ'נציגי מכירות'. המסנן השני מחלץ נתוני 'לקוחות' ממדריד שבספרד. לבסוף, השיטה PopulatePartial יוצרת סינכרון בין העותק המלא שצוין באמצעות השרשור של path ו- sourceName לבין העותק החלקי החדש. רק שתי טבלאות קולטות רשומות (באפשרותך להוסיף מסננים כדי לאכלס טבלאות נוספות בעותק החלקי).

סינכרון עותקים משוכפלים

שתי השגרות הבאות יוצרות סינכרון בין עותקים בתרחישי שכפול טיפוסיים. שתיהן מפעילות שגרות ADO בסיסיות. השיגרה synchNorthwindFooToAdd מוסיפה רשומה חדשה בטבלה Employees שבעותק Northwind.mdb. לאחר מכן היא יוצרת סינכרון בין Northwind ל- foo כדי להפיץ את הרשומה החדשה אל foo.mdb. השיגרה השנייה, synchFooNorthwindToDelete, מוחקת את רשומת העובד החדשה מ- foo, ולאחר מכן יוצרת סינכרון בין foo ל- Northwind כדי למחוק את הרשומה גם מהעותק Northwind.

```
Sub synchNorthwindFooToAdd()
Dim rep1 As JRO.Replica
Dim cnn1 As New ADODB.Connection
Dim rst1 As ADODB.Recordset
' Open connection to Northwind and set reference for Northwind as a replica.
    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=c:\Program Files\Microsoft Office\" & _
        "Office\Samples\Northwind.mdb"
    Set rep1 = New JRO.Replica
    rep1.ActiveConnection = cnn1

' Add a new employee to Northwind.
    Set rst1 = New ADODB.Recordset
    rst1.Open "Employees", cnn1, adOpenKeyset, adLockOptimistic, adCmdTable
    rst1.AddNew
        rst1.Fields("FirstName") = "Rick"
        rst1.Fields("LastName") = "Dobson"
        rst1.Fields("BirthDate") = Date - 1
' When it comes to learning about computers and my faith
' in the Lord, I am always newly born.
    rst1.Update
' Synchronize Northwind with its full replica (foo.mdb).
    rep1.Synchronize "c:\My Documents\foo.mdb", _
        jrSyncTypeImpExp, jrSyncModeDirect

End Sub

Sub synchFooNorthwindToDelete()
Dim rep1 As JRO.Replica
Dim cnn1 As New ADODB.Connection, cmd1 As ADODB.Command
' Open connection to foo and set reference to foo as a replica.
    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=c:\My Documents\foo.mdb"
    Set rep1 = New JRO.Replica
    rep1.ActiveConnection = cnn1
```

```
' Execute command to remove an employee from foo.mdb.
    Set cmd1 = New ADODB.Command
    With cmd1
        .ActiveConnection = cnn1
        .CommandText = "DELETE Employees.* FROM Employees" & _
            " WHERE LastName='Dobson'"
        .CommandType = adCmdText
        .Execute
    End With
' Synchronize foo with its design master (Northwind.mdb).
    rep1.Synchronize "c:\Program Files\Microsoft Office\" & _
        "Office\Samples\Northwind.mdb", jrSyncTypeImpExp, jrSyncModeDirect
End Sub
```

השיגרה synchNorthwindFooToAdd משתמשת ב- ADO כדי להוסיף רשומה לטבלה באחד העותקים, ולהפיצה אל טבלה מקבילה בעותק אחר. השיגרה פותחת בהצהרה על האובייקטים Replica, Connection ו- Recordset. לאחר מכן היא יוצרת חיבור עם מסד הנתונים Northwind ומציבה את המאפיין ActiveConnection של העותק בחיבור. היא יוצרת מופע של אובייקט Recordset באותו חיבור שבו נמצא העותק. לאחר מכן היא מוסיפה לטבלה רשומה של עובד בשם Rick Dobson. השיגרה מסתיימת בהחלת השיטה Synchronize על העותק Northwind, ומציינת את foo כעותק שעמו יש להחליף עדכונים. בשלב האחרון, רשומת העובד החדשה עוברת מ- Northwind ל- foo.

השיגרה synchFooNorthwindToDelete מוחקת את הרשומה החדשה מ- foo. כמו כן היא מחזירה את Northwind למצבו הקודם על ידי פעולת סינכרון בין foo ל-Northwind. שיגרה זאת משתמשת באובייקט Command כדי להשמיט עובד ששם משפחתו Dobson מהטבלה Employees בעותק foo. לאחר ביצוע הפקודה, היא מחילה את השיטה Synchronize על העותק foo כדי להפיץ את המחיקה לעותק Northwind.

עבודה עם עותקים משוכפלים הנמנעים ממחיקות

אין קושי ליצור עותק משוכפל עם מניעת מחיקות, אולם עליך ליצור אותו דרך ממשק המשתמש. בחר **כלים** (Tools), **שכפול** (Replication), **יצירת עותק משוכפל** (Create Replica), ובחר בתיבת הסימון **מניעת מחיקות** (Prevent Deletes) בתיבת הדו-שיח שתיפתח על המסך. עותק משוכפל מסוג זה מאפשר ליישום להפיץ עותק שאינו מאפשר מחיקה ישירה של פריטים מתוכו. אמנם ניתן להשיג תוצאה זהה באמצעות הגדרות האבטחה של Access, אולם הרבה יותר פשוט לבחור את תיבת הסימון הנזכרת לעיל. בעוד שהמשתמש בעותק עם מניעת מחיקות אינו יכול למחוק רשומות במישרין, העותק המשוכפל יכול לקבל עדכוני מחיקה מעותקים אחרים. אחד השימושים לעותק משוכפל מסוג זה הוא לסייע בידי מנהלים למנוע מחיקת רשומות לפני שמירתן הנאותה בארכיב.

שלוש השגרות הבאות מטפלות בעותקים עם מניעת מחיקות בעזרת ADO. השיגרה SynchAddToFoo2 מוסיפה מתוך העותק foo.mdb רשומת עובד חדשה לעותק foo2.mdb. שם העובד הוא Rick Dobson. השיגרה השנייה, TryToDeleteFromFoo2, מנסה למחוק את הרשומה הזאת במישרין מהעותק foo2.mdb. הודעת השגיאה המוצגת בתרשים 11.3 מראה כיצד Access מגיב לפקודה Execute הכלולה בשיגרה. אם מתעורר צורך בהגבלת הוספות פרט להגבלת מחיקות, תוכל להציב בארגומנט Updatability של השיטה CreateReplica את הערך jrRepUpdReadOnly. אין צורך בממשק המשתמש כדי לבצע פעולה זאת. השיגרה השלישית, synchFooFoo2ToDelete מוחקת את Rick Dobson מהעותק foo.mdb, ולאחר מכן מפיצה את המחיקה אל העותק foo2.mdb. למרות שהמשתמש אינו יכול למחוק רשומה במישרין מתוך עותק עם מניעת מחיקות, מנהל מערכת יוכל להפיץ מחיקה בעזרת פעולה של סינכרון.

```
Sub SynchAddToFoo2()
Dim rep1 As JRO.Replica
Dim cnn1 As New ADODB.Connection
Dim rst1 As ADODB.Recordset
' Open connection to foo and set reference to foo as a replica.
   cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=c:\My Documents\foo.mdb"
   Set rep1 = New JRO.Replica
   rep1.ActiveConnection = cnn1

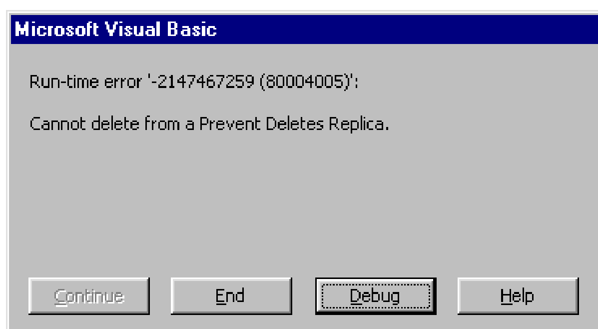
' Add a new employee to foo.
   Set rst1 = New ADODB.Recordset
   rst1.Open "Employees", cnn1, adOpenKeyset, adLockOptimistic, adCmdTable
   rst1.AddNew
      rst1.Fields("FirstName") = "Rick"
      rst1.Fields("LastName") = "Dobson"
      rst1.Fields("BirthDate") = Date - 1
   rst1.Update

' Synchronize foo with foo2.mdb.
   rep1.Synchronize "c:\My Documents\foo2.mdb", _
      jrSyncTypeImpExp, jrSyncModeDirect
End Sub

Sub TryToDeleteFromFoo2()
Dim rep As JRO.Replica
Dim cnn1 As New ADODB.Connection, cmd1 As ADODB.Command
' Open connection to foo2 and set a reference to it as a replica.
   cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=c:\My Documents\foo2.mdb"
   Set rep1 = New JRO.Replica
   rep1.ActiveConnection = cnn1
```

```
' Execute command to remove employee from foo2.mdb;
' it fails because Foo2 is a Prevent Deletes replica.
    Set cmd1 = New ADODB.Command
    With cmd1
        .ActiveConnection = cnn1
        .CommandText = "DELETE Employees.* FROM Employees" & _
            " WHERE LastName='Dobson'"
        .CommandType = adCmdText
        .Execute
    End With
End Sub

Sub synchFooFoo2ToDelete()
Dim rep1 As JRO.Replica
Dim cnn1 As New ADODB.Connection, cmd1 As ADODB.Command
' Open connection to foo and set reference to foo as a replica.
    cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=c:\My Documents\foo.mdb"
    Set rep1 = New JRO.Replica
    rep1.ActiveConnection = cnn1
' Execute command to remove an employee from foo.mdb.
    Set cmd1 = New ADODB.Command
    With cmd1
        .ActiveConnection = cnn1
        .CommandText = "DELETE Employees.* FROM Employees" & _
            " WHERE LastName='Dobson'"
        .CommandType = adCmdText
        .Execute
    End With
' Synchronize foo with its design master (Northwind.mdb).
    rep1.Synchronize "c:\My Documents\foo2.mdb", _
        jrSyncTypeImpExp, jrSyncModeDirect
End Sub
```



תרשים 11.3: הודעת השגיאה המוצגת בעקבות ניסיון למחוק רשומה מתוך עותק המודגם עם התכונה **מנע מחיקות**

עבודה עם מאפייני עותק משוכפל

בדיקת המאפיינים של עותק משוכפל תסייע בידך להבין את התנהגותה של קבוצת עותקים משוכפלים. לדוגמה, אם ידוע לך סוג העותק המשוכפל, תדע גם אם אותו עותק ישתף שינויי סכימה עם עותקים משוכפלים אחרים. המאפיין Priority (קדימות) של עותק משוכפל קובע איזה משני עותקים 'ינצח' כאשר יש התנגשות בין שני עותקים. ככלל, העותק בעל הקדימות הגבוהה יותר ינצח. שלוש השגרות הבאות מדפיסות את המאפיינים של עותק משוכפל וחושפות תוך כדי כך ערכי מאפיינים.

הואיל ולא קיים אוסף Replicas, קרוב לוודאי שתציין לפי סדר רשימה ובה מספר עותקים משוכפלים שברצונך לעבד, במיוחד כאשר העותקים המשוכפלים משתייכים לקבוצות עותקים משוכפלים שונות. השיגרה callPrintTypePriority מגדירה משתני path ו- replicaName עבור כל עותק משוכפל שהיא מבקשת לבדוק, ולאחר מכן היא מפעילה שתי שגרות משניות. אחת מהן (printReplicaType), מחזירה נתונים אודות סוג העותק המשוכפל, ואילו השנייה (printPriority) מחזירה את ערך המאפיין Priority של העותק המשוכפל. נוסף לארגומנטים Path ו-replicaName, קבוצת השגרות מחליפה ערכים עם ארגומנט ששמו Exist. זהו משתנה בוליאני המציין אם קובץ מסוים קיים בנתיב נתון. אם הקובץ לא קיים, ברור מאליו שאין לו ערך קדימות.

```
Sub callPrintTypePriority()
On Error GoTo TypeTrap
Dim repMaster As New JRO.Replica
Dim path As String, replicaName As String
Dim Exist As Boolean
' Assign Boolean for file existing.
   Exist = True

' Assign path and replica names, then
' call procedures for printing type and priority.
   path = "C:\Program Files\Microsoft Office\Office\Samples\"
   replicaName = "Northwind.mdb"
' If file does not exist, the next line catches it.
   repMaster.ActiveConnection = path & replicaName
   printReplicaType replicaName, repMaster.ReplicaType, Exist
   printPriority replicaName, path, Exist
   replicaName = "Copy of Northwind.mdb"
   repMaster.ActiveConnection = path & replicaName
   printReplicaType replicaName, repMaster.ReplicaType, Exist
   printPriority replicaName, path, Exist
   replicaName = "Northwind2.mdb"
   repMaster.ActiveConnection = path & replicaName
   printReplicaType replicaName, repMaster.ReplicaType, Exist
   printPriority replicaName, path, Exist
```

```
    path = "C:\My Documents\"
    replicaName = "foo.mdb"
    repMaster.ActiveConnection = path & replicaName
    printReplicaType replicaName, repMaster.ReplicaType, Exist
    printPriority replicaName, path, Exist

    path = "C:\Program Files\Microsoft Office\Office\Samples\"
    replicaName = "Partial of Northwind.mdb"
    repMaster.ActiveConnection = path & replicaName
    printReplicaType replicaName, repMaster.ReplicaType, Exist
    printPriority replicaName, path, Exist

    path = "C:\My Documents\"
    replicaName = "Partial of foo.mdb"
    repMaster.ActiveConnection = path & replicaName
    printReplicaType replicaName, repMaster.ReplicaType, Exist
    printPriority replicaName, path, Exist
    replicaName = "DMBackup.mdb"
    repMaster.ActiveConnection = path & replicaName
    printReplicaType replicaName, repMaster.ReplicaType, Exist
    printPriority replicaName, path, Exist

    path = "C:\My Documents\"
    replicaName = "foo2.mdb"
    repMaster.ActiveConnection = path & replicaName
    printReplicaType replicaName, repMaster.ReplicaType, Exist
    printPriority replicaName, path, Exist
TypeExit:
    Exit Sub

TypeTrap:
    If Err.Number = -2147467259 And _
        Left(Err.Description, 19) = _
        "Could not find file" Then
        Exist = False
        Resume Next
    Else
        Debug.Print Err.Number, Err.Description
        Resume TypeExit
    End If
End Sub
```

```
Sub printReplicaType(repName As String, _
    typeNumber As Integer, Exist As Boolean)

    If Exist Then
' Decode replica type enumeration constants or...
        Select Case typeNumber
            Case jrRepTypeNotReplicable
                Debug.Print repName & " is not replicable."
            Case jrRepTypeDesignMaster
                Debug.Print repName & " is a Design Master."
            Case jrRepTypeFull
                Debug.Print repName & " is a Full Replica."
            Case jrRepTypePartial
                Debug.Print repName & " is a Partial Replica."
        End Select
    Else
' print that file does not exist.
        Debug.Print repName & " does not exist."
    End If

End Sub

Sub printPriority(replicaName As String, path As String, Exist)
Dim repMasterP As New JRO.Replica
' Print priorty and reset Exist.
    If Exist = True Then
' Assign connection for replica.
        repMasterP.ActiveConnection = path & replicaName

        If repMasterP.ReplicaType <> jrRepTypeNotReplicable Then
' Print priority for replicas.
            Debug.Print "It's priority is " & repMasterP.Priority & "."
        Else
' Print message for no replica.
            Debug.Print "Therefore, it has no priority."
        End If
    Else
' Print message for file does not exist.
        Debug.Print "Therefore, it has no priority."
    End If
    Debug.Print

    Exist = True
End Sub
```

השיגרה callPrintTypePriority משתמשת בלכידת שגיאות כדי לקבוע אם קובץ נתון קיים, וכדי להגיב כיאות אם מתברר שאינו קיים. כאשר היא מנסה להקצות את המאפיין ActiveConnection של מופע עותק משוכפל לקובץ שאינו קיים, רכיב השכפול של Jet מחזיר את קוד Err המקובל (-2147467259), בתוספת משפט תיאורי. מלכודת השגיאות משווה את המשפט עם קוד Err של השכפול ב- Jet. אם הלוגיקה ללכידת שגיאות קובעת שהקובץ אכן אינו קיים, המלכודת מציבה ערך False בארגומנט Exist, וממשיכה. שתי שגרות המשנה, printReplicaType ו- printPriority, מפרשות את הערך False שהוצב ב- Exist ומגיבות בהתאם.

השיגרה callPrintTypePriority שולחת את ערך המאפיין ReplicaType עבור עותק משוכפל כאשר היא מפעילה את השיגרה printReplicaType. השיגרה printReplicaType בודקת את הערך של Exist לפני שהיא מנסה לפענח את ערך המאפיין ReplicaType. אם Exist מכיל ערך False, השיגרה רק תציג בחלון Immediate (מיידי) הודעה על כך שהקובץ אינו קיים. אם Exist מכיל True, משפט SelectCase יפענח את המאפיין ReplicaType.



הערה:

השיגרה printReplicaType מייצגת ערכי ReplicaType בעזרת שמות הקבוע ReplicaTypeEnum ב- JRO. תוכל לעיין בשמות אלה בעזרת **Object Browser** (סורק האובייקטים). הקלד ReplicaType בתיבה **Search text**, ולחץ על הלחצן **Search**. מהקבוצה המוחזרת, בחר את המחלקה ReplicaTypeEnum עבור ספריית JRO (זהירות: בספריית DAO קיימת מחלקת ReplicaTypeEnum שונה). בחירה זאת תגרום להצגת רשימת שמות הקבוע וערכי ReplicaType שלהם.

עיצוב השיגרה printPriority שונה במקצת מזה של השיגרה printReplicaType. השיגרה printPriority יוצרת בתוך עצמה מופע של עותק משוכפל. לאחר מכן היא ממצה ערך מאפיין עבור העותק המשוכפל. נוסף לבדיקת הערך המוצב במשתנה Exist, היא בודקת גם את המאפיין ReplicaType במופע העותק המשוכפל. אם מתברר שהעותק אינו ניתן לשכפול (לפי הערך המוחזר jrRepTypeNotReplicable), השיגרה מדפיסה הודעה שלפיה אין ערך קדימות עבור הקובץ. אחרת, השיגרה מדפיסה את הערך המוחזר של המאפיין Priority במופע העותק המשוכפל. בטרם תחזיר את השליטה לשיגרה callPrintTypePriority, השיגרה printPriority מגדירה את הערך של Exist כ- True כדי לעבד את העותק המשוכפל הבא.

דחיסה והצפנה של עותקים משוכפלים

ספריית JRO תומכת ביותר מעותק משוכפל אחד. לדוגמה, ניתן לדחוס ולהצפין קבצים, וכן ניתן לרענן את מטמון הזיכרון. שתי השגרות הבאות יוצרות גיבוי של עותק משוכפל על ידי דחיסתו והצפנת העותק הדחוס. גישה זאת מתאימה במיוחד

כאשר שולחים באינטרנט קובץ המכיל מידע רגיש. הדוגמה דוחסת את הקובץ Northwind.mdb אל תוך Northwind2.mdb. אם רצונך בכך, תוכל לציין נתיבים שונים עבור Northwind ו- Northwind2.

```
Sub callCompactADB()
    compactADB "Northwind.mdb", "Northwind2.mdb", _
        "C:\Program Files\Microsoft Office\Office\Samples\"
End Sub

Sub compactADB(oName As String, cName As String, _
    opath As String, Optional cpath As String)
Dim je As New JRO.JetEngine
Dim strIn As String, strOut As String
' Is optional path specified?
    If cpath = "" Then
        cpath = opath
    End If

    strIn = opath & oName
    strOut = cpath & cName
    deleteFile strOut

    je.CompactDatabase _
    "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=" & strIn, _
    "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        "Data Source=" & strOut & ";" & _
    "Jet OLEDB:Encrypt Database=True"
End Sub
```

12

בניית פתרונות עם MSDE ופרוייקטי Access

Microsoft Access 2000 מציג שני חידושים בעבודה עם מסדי נתונים של Microsoft SQL Server. החידוש הראשון הוא MSDE (Microsoft Data Engine), מנגנון הנתונים של Microsoft. MSDE מאפשר אחסון של נתונים מקומיים בתבנית מסדי נתונים של SQL Server. בדומה למסדי נתונים של SQL Server, מסדי נתונים של MSDE יכולים לשמש כמערכות עורפיות בפתרונות העוסקים בקבצים מסוג adp.*. הפיתוח עם MSDE יקל עליך את המעבר מפתרונות המשרתים צרכים מקומיים, מחלקתיים, לפתרונות מורחבים המשרתים את צורכיהם של ארגונים שלמים ובהם אלפי משתמשים. החידוש השני הוא קבצים מסוג adp.* (אותיות הסיומת adp מייצגות את המילים Access Data Project). ב- Microsoft מכנים יישומים שפותחו בעזרת קבצי adp.* **פרויקטים של Access**. בפרויקטים של Access נעשה שימוש בטפסים, דוחות, מודולים ואפילו פקודות מאקרו, כמקובל, אולם עם מסד נתונים של SQL Server במקום מסד נתונים Jet. קובץ מסד הנתונים מהסוג החדש אינו מכיל טבלאות או שאילתות של Access Microsoft, אולם הוא מאפשר גישה אל טבלאות, תצוגות ושגרות מאוחסנות של SQL Server.

בפרק זה תלמד כיצד להתקין את מנגנון הנתונים של Microsoft ולנהל אותו, וכיצד לפתח יישומים מותאמים אישית בעזרת פרויקטים של Access. כמו כן תלמד כיצד לעבוד עם טבלאות, תצוגות, שגרות מאוחסנות, דוחות וטפסים. דוגמאות הקוד יבהירו כיצד ליצור אוטומציה של היישומים בפרויקטים של Access בעזרת ADO (ActiveX Data Objects) ו- Microsoft VBA (הפרקים 3 ו-4 עוסקים בנתיבי OLE DB לתבניות נוספות של מסדי נתונים מרוחקים פרט ל- SQL Server).

מנגנון הנתונים של Microsoft (MSDE)

תוכל לנצל את מנגנון הנתונים של Microsoft לבניית פתרונות על גבי מחשבים אישיים בעלי מעבד יחיד, המופעלים במערכות הפעלה Microsoft Windows 9x או Microsoft Windows NT. פתרונות המופעלים על גבי MSDE מתאימים לקבוצות עבודה קטנות, אולם ניתן בנקל להעביר פתרונות MSDE גם ל- Microsoft SQL Server 7, ובאופן זה להרחיבם לאשכולות מרובי-מעבדים המופעלים במערכות ההפעלה המתקדמות ביותר של Windows. מערכות מתקדמות אלה כוללות את Windows NT Server Enterprise Edition, ומערכות הפעלה מתקדמות יותר מסדרת Windows 2000.

פתרונות MSDE תואמים לחלוטין את אלה שפותחו עבור SQL Server 7. תוכל ליצור דגם אב של פתרון בעזרת MSDE וקבצי adp.* במחשב המקומי, ולאחר מכן להעביר את הטבלאות, התצוגות והשגרות המאוחסנות של מסד הנתונים ל- SQL Server לצורך בדיקה, ליטוש ופרסום. ניתן להפיץ בין הבוחנים והמשתמשים את קבצי adp.* ולשנות את קישור הנתונים שלהם כך שיצביע אל מסד הנתונים של SQL Server במקום אל מערכת האב המבוססת על MSDE. יישומים מסוג זה מתאימים לצרכיו של ארגון, ולא רק לצרכים מחלקתיים.

MSDE לעומת Jet

בעיקרם, פתרונות Access המבוססים על Jet קלים יותר לניהול, מידת התאימות שלהם עם פתרונות שנבנו בעזרת גרסאות קודמות של Access גדולה יותר, והם גוזלים פחות משאבים. בניית פתרונות Access המבוססים על MSDE מחייבת בקיאות מסוימת בסוגי הנתונים החדשים, בכללים השונים לפיתוח שאילתות, ובטכניקות של ניהול מסדי נתונים. לעומת זאת MSDE מציע אפשרויות שחזור נתונים מעולות, רישום טרנזקציות מוכלל, אבטחת Windows NT משולבת ופוטנציאל לקיבולת מסד נתונים אדירה (בעת המרה ל- SQL Server 7). תוכל לנצל את ממשק המשתמש של Access לבניית טפסים ודוחות עבור פתרונות המבוססים על כל אחד משני המנגנונים הללו. בשל העיצוב במתכונת שרת-לקוח הטבוע בפתרונות MSDE, יהיה עליך לציית למספר כללים בהפעלת MSDE והממשק למשתמש. תוכל לתכנת פתרונות לשני מנגנוני מסדי הנתונים גם בעזרת VBA ו- ADO.

הן MSDE והן Jet תומכים בשלמות קשרים מוצהרת, דבר שמאפשר הצהרה על יחסים בין טבלאות באמצעים גרפיים. עם זאת, בעוד ש-Jet תומך בעדכונים ומחיקות בהתאם להיררכית קשרים, ברמת המנגנון, עם MSDE חובה לממש תכונות אלה ב- SQL באמצעות גורמים (triggers) טבועים בקוד. ניתן ליצור תבנית גורם לטבלה בלחיצת עכבר ימנית על הטבלה הרצויה בחלון **מסד נתונים** (Database), ובחירה באפשרות **גורמים** (Triggers). לאחר מכן יש ללחוץ על **חדש** (New) כדי לפתוח את התבנית. ב- SQL Server Books Online תמצא כללי תחביר, דוגמאות ורקע כללי על פיתוח גורמים).

אם דרוש לך רק שחזור מלא מתוך קובץ גיבוי, בחר **כלים** (Tools), **עזרי מסד נתונים** (Database Accessories), **שחזור** (Restore) בפרויקט Access. פעולה זאת דומה לשמירת עותקי גיבוי של קבצי מסד נתונים ב- Jet ושחזור הקבצים מתוך העותקים. רק MSDE תומך בשחזור עד לנקודת זמן נתונה מתוך קובץ יומן. עליך לתכנת שחזור מסוג זה בעזרת Transact-SQL, שיש לו תמיכה בעזרה המקוונת של Access 2000. העזרה המקוונת כוללת דוגמאות script ותחביר לשחזור נתונים בטבלה עד לנקודת זמן נתונה מתוך קובץ היומן של הטבלה.



הערה:

אם בנסיבות מסוימות מוטל עליך לנהל אוסף של משימות המחייבות שימוש ב- Transact-SQL, רצוי להשיג עותק מלא של SQL Server 7 (במקום חבילת MSDE החופשית המכילה בעיקרה קוד זהה של מנגנון מסד נתונים). SQL Server Enterprise Manager מכיל אוסף גדול של אשפים וכן ממשק גרפי למשתמש, המבטל את הצורך בקוד של Transact-SQL לביצוע משימות כמו גיבוי ושחזור.

קבצי מסד נתונים ב- Jet הם ככלל קטנים מקבצים שווי-ערך המבוססים על MSDE. גודלו של קובץ Northwind.mdb עומד על כ- 2MB. היקף זה כולל את כל טבלאות מסד הנתונים, שאר האובייקטים של מסד הנתונים, והקוד. גירסת Northwind המבוססת על MSDE כוללת קובץ מסד נתונים (*.mdf) וקובץ של פרויקט Access (*.adp). גודלם של קבצים אלה עומד על כ- 3.7MB. פתרונות MSDE מנהלים באופן אוטומטי קבצי יומן המסייעים במקרי שחזור. עם זאת, על מסדי נתונים ב- Jet להעביר את מסדי הנתונים שלהם ברחבי הרשת כשמדובר בפתרונות מרובי-משתמשים. לעומת זאת פתרונות המבוססים על MSDE מבצעים את כל עיבודי מסד הנותנים בשרת, ומעבירים רק את הקבוצות המוחזרות ברחבי הרשת. דבר זה עשוי להפחית את נפח התעבורה ברשת, ולהאיץ את מהירות הביצוע. הן Jet והן MSDE מאפשרים מסדי נתונים בנפח של עד 2GB, אולם פתרונות MSDE ניתנים להמרה בנקל למערכות SQL Server 7 ויכולים לתמוך במסדי נתונים בנפח העולה על 1 מיליון טרה-בית (ט״ב). רמת הביצועים של SQL Server 7 עולה ככל שעולה מספר המעבדים במחשבים שבהם הוא מופעל, אולם רמת הביצועים של Jet אינה משתנה באופן משמעותי בעקבות תוספת של מעבדים.



הערה:

בדומה ל- SQL Server 7 הוותיק יותר, MSDE מיועד לעבודה עם קבוצות עבודה קטנות.

Jet ו- MSDE נועדו, כל אחד, למטרות משלו. MSDE הוא הכלי האידיאלי עבורך אם בכוונתך להשתמש ב- SQL Server בגרסאות עתידיות של היישום. כאשר MSDE פועל במערכות Windows NT, הוא מעניק אבטחה משולבת של מערכת ההפעלה ומסד הנתונים, דבר העשוי להפחית את העומס בתחום ניהול האבטחה. כמו כן MSDE

מאפשר שחזור לנקודת זמן נתונה. לעומתו, Jet מתאים יותר ליישומים שאינם יישומי משימה קריטיים, אשר השיקול המכריע בהם הוא פשטות הפיתוח. פתרונות Jet גוזלים פחות מקום ובמקרים מסוימים הם עדיפים, אם אתה כפוף למגבלות של זיכרון או מקום פנוי בדיסק. הואיל ומנגנון Jet המשווק עם Access 2000 עודכן במידה שולית בלבד, רמת התאימות הגבוהה ביותר שלו היא מול Access 97 וגרסאות קודמות.

התקנה והגדרת התצורה של MSDE

MSDE הוא מנגנון מסד נתונים אופציונלי למפתחי ומשתמשי Access. תוכל להתקין אותו על ידי הפעלת הקובץ SetupSQL.exe מתוך תקליטורי Office 2000. אין צורך בהכנות מיוחדות לפני התקנת MSDE במערכות הפעלה Windows 9x. לעומת זאת, על משתמשי Windows NT 4 לבצע שתי פעולות כהכנה להתקנת MSDE. תחילה עליהם להתקין את Service Pack 3, ולאחר מכן להפעיל את הקובץ hotfix.exe מתוך תקליטורי Office 2000. ב- Windows NT 4 עם Service Pack 4 אין צורך בפעולות מכינות לקראת התקנה.

לאחר שהתקנת את MSDE, עליך להפעיל את המנגנון בטרם תוכל להשתמש בו. במחשבים עם מערכות הפעלה Windows 9x, בצע את הפעולות הבאות:

1. בתפריט **התחלה** (Start) בחר באפשרות **תוכניות** (Programs).
2. בחר באפשרות **Msde** ולאחר מכן בפריט **Service Manager** (מנהל השירותים).
3. בתיבת הרשימה הנפתחת **Services** (שירותים), בחר באפשרות **MSSQLServer**, ולחץ על **Start/Continue** (התחל/המשך).

תוכל לבחור את תיבת הסימון **Auto-Start Service** אם אתה מעוניין בהפעלה אוטומטית של MSDE בכל אתחול של מערכת ההפעלה. אם לא בחרת באפשרות זאת, תיאלץ לחזור על הפעולות המתוארות לעיל כל פעם שתשתמש ב- MSDE.

משתמשי Windows NT יכולים להפעיל את MSDE כמו כל שירות אחר של Windows NT. בלוח הבקרה, לחץ לחיצה כפולה על הסמל **Services**, ולאחר מכן בחר באפשרות **MSDE Services**. לחץ על **Start** בתיבת הדו-שיח **Services** כדי להפעיל את השירות. כעת לחץ על **Startup** ובחר אחת מהאפשרויות בפריט **Startup Type** (**Automatic**, **Manual**, או **Disabled**).

לאחר ההתקנה וההפעלה, יופיע בשורת המשימות של Windows סמל המייצג את מנהל השירותים ב- SQL Server. השתמש בסמל זה כדי להפעיל, להשהות או לעצור את MSDE. השתמש בו גם לפתיחת **Service Manager** כדי שתוכל לבצע פונקציות דומות עבור Microsoft Distributed Transaction Coordinator וסוכן Microsoft SQL Server.

מקובל לדרוש מהמשתמש להזין זיהוי וסיסמה בטרם יורשה לפתוח קובץ עבור מסד נתונים שרת-לקוח. MSDE מאפשר לך לנהל את אבטחת הכניסה בעזרת חשבונות פרטיים וקבוצתיים עבור מסדי נתונים מוגדרים, או בעזרת אבטחת pass-through משולבת הלקוחה ממערכת Windows NT. ערכי ברירת המחדל של MSDE הם sa כזיהוי משתמש, ותו ריק כסיסמה. כדי ליצור חשבונות אבטחה לגישה אל מסדי נתונים נפרדים, בחר **כלים** (Tools), **אבטחה** (Security), **אבטחת מסד נתונים** (Database Security). משתמשי Windows 9x יכולים להשתמש בחשבונות אבטחה של MSDE רק למסדי נתונים נפרדים. התפקידים של מסדי נתונים ב- MSDE דומים לתפקידי הקבוצות ב- Access. מסד נתונים יחיד יכול להשתייך לשני תפקידים שונים, או יותר, והוא יורש את רמת האבטחה הנמוכה ביותר מאלה החלות בתפקידים השונים שאליהם הוא משתייך.

כאשר מפעילים את MSDE מתוך Windows 9x, הוא תומך בפרוטוקול TCP/IP ו ב- Multi-Protocol, אך לא ב- Named Pipes. עם זאת, רוב הלקוחות מנסים להתחבר לשרת MSDE באמצעות Named Pipes כברירת מחדל. אם אתה מעוניין שמחשב-לקוח ייצור כינוי עבור שרת MSDE המתחבר באמצעות TCP/IP או באמצעות Multi-Protocol, בחר בפקודה Client Network Utility מתוך תפריט המשנה **Msde**, לאחר לחיצה על **התחל**.

פרויקטי Access

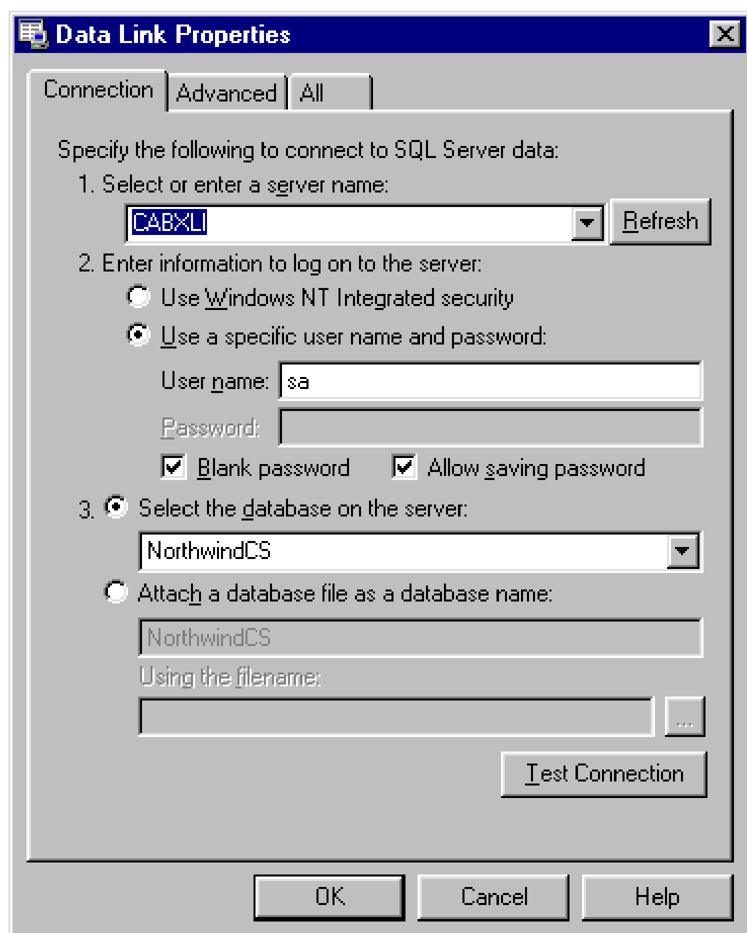
בעזרת פרויקטים של Access עם MSDE או עם SQL Server, תוכל לפתח יישומי שרת-לקוח באותה קלות שאתה מפתח יישומי שרת קבצים. הממשק של פרויקטי Access חושף תשעה סוגים בסיסיים של אובייקטי יישומים. טבלאות, תצוגות, שגרות מאוחסנות ודיאגרמות של מסדי נתונים מיועדים לקטלוג של מסד נתונים MSDE או SQL Server; אלה נשמרים בקובץ מסד הנתונים. טפסים, דוחות, דפי גישה לנתונים (לפיתוח תכני Web), פקודות מאקרו ומודולים נשמרים בקובץ adp.*. פרויקט Access מבצע תיאום עם מסד הנתונים שרת-לקוח באמצעות חיבור OLE DB. קובץ מסד הנתונים שהוא בדרך כלל קובץ מסוג mdf.*, וקובץ adp.* – הם שני החלקים המרכיבים את פתרון השרת-לקוח.

חיבור פרויקט Access עם מסד נתונים

תוכל לחבר קובץ adp.* עם מסד נתונים מאחד הסוגים הבאים:

- קובץ מסד נתונים SQL Server 7 במערכת Windows 9x או Windows NT (עם Service Pack 4 ומעלה).
- קובץ MSDE במערכת Windows 9x או Windows NT.
- SQL Server 6.5 במערכת Windows NT (Service Pack 5 ומעלה).

כדי לפקח על החיבור עם מסד הנתונים שרת-לקוח, או כדי לאפס אותו, בחר באפשרות **התקשרות** (Connection) מתוך התפריט **קובץ** (File). בתיבת הדו-שיח **Data Link Properties** (המוצגת בתרשים 12.1, להלן), בחר בשרת של מסד נתונים, סוג אבטחה ושם מסד הנתונים. בדרך כלל, שם השרת של מסד הנתונים הוא שם המחשב שבו מופעל השרת.



תרשים 12.1: תיבת הדו-שיח **Data Link Properties** (מאפייני קישור הנתונים)

התרשים מציג חיבור אל מסד נתונים MSDE הפועל במחשב ששמו CABXLI. שום פרט בתיבת הדו-שיח אינו מציין במפורש שמדובר במסד נתונים MSDE, ולא במסד נתונים SQL Server. הואיל והמחשב CABXLI מבוסס על מערכת ההפעלה Windows 9x, אין באפשרותו להפעיל אבטחה משולבת של Windows NT. תיבת הדו-שיח **Data Link Properties** משתמשת בערכי ברירת המחדל של נתוני הקישוריות הייחודיים למשתמש. לאחר שהוקם הקשר עם שרת מסד הנתונים, תיבת הרשימה הנפתחת מציגה את כל מסדי הנתונים שנמצאים בשרת. אחרי שתבחר בפריטים

הרצויים, קובץ adp.* יאכלס באובייקטים את הטבלאות שלו ואת התצוגות, השגרות המאוחסנות ואוספי התרשימים של מסדי הנתונים. הגדרות אלה נשארות בתוקפן בין מופעים שונים של פרויקט Access נתון.

לימוד ממסדי הנתונים NorthwindCS ו- Pubs

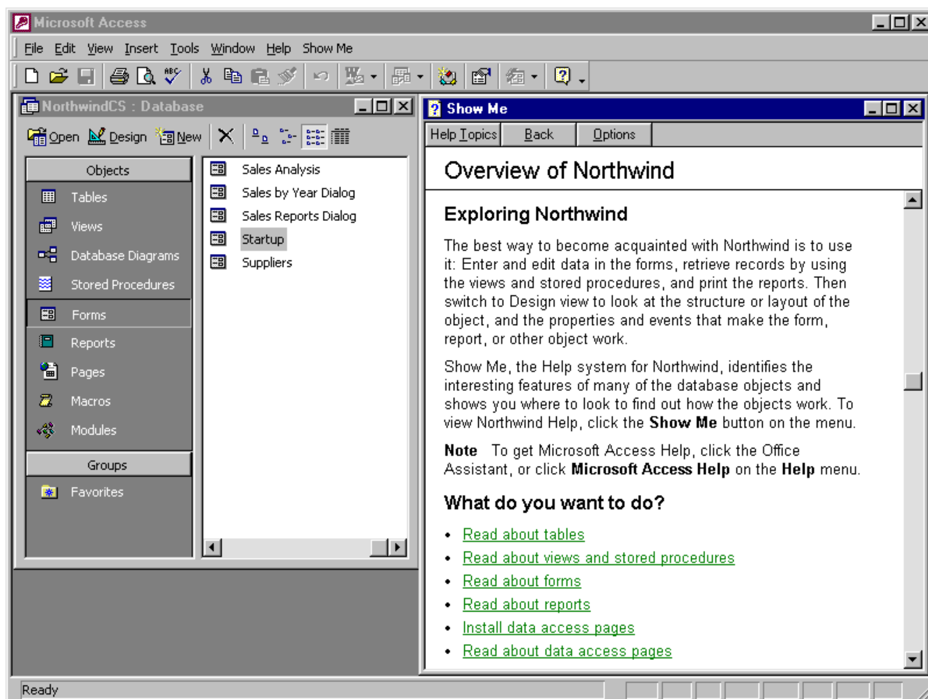
Access 2000 משווק הן עם שרת הקבצים והן עם גרסת שרת-לקוח של יישום ההדגמה Northwind. גירסת השרת-לקוח מוגדרת כפרויקט Access. מסד הנתונים Pubs אינו משווק עם Access 2000, אולם מזה זמן רב הוא נחשב למסד הדוגמה הסטנדרטי של SQL Server. אם אתה מפעיל פרויקטים של Access עם מנהל מסד נתונים SQL Server, אין כמעט ספק שדוגמה זאת תעמוד לרשותך. תוכל לפתוח אותה בתוך פרויקט Access. דוגמאות ADO ודוגמאות מסדי נתונים רבות אחרות בקבצי העזרה המקוונת של Access 2000 מכילות הפניות למסד נתונים זה. לכן, תוכל להרחיב את היכרותך עם התכונות החדשות של מסדי נתונים ב- Access 2000 באמצעות ההיכרות עם מסד הנתונים Pubs.

שימוש בפרויקט ובמסד הנתונים של NorthwindCS

הקובץ NorthwindCS.adp מועתק אל המחשב שלך בעת ההתקנה של Access Samples (מסדי נתונים לדוגמה של Access). זהו פרויקט Access בעל רכיבים תפקודיים דומים ונתונים זהים לאלו שבקובץ Northwind.mdb הקלאסי. Access 2000 מעמיד לרשותך את שני הקבצים. כדי להשתמש ב-NorthwindCS, עליך להתקין תחילה את MSDE במחשב, או ליצור חיבור עם מנהל מסד נתונים SQL Server 7, מכיון שקבצי מסד הנתונים של Northwind משווקים עם שני מוצרים אלה.

כאשר תפעיל את NorthwindCS.adp לראשונה, קטע script מתאים יברר אם MSDE מותקן במחשב שלך. אם כן, תתבקש להורות ל-script אם ברצונך לטעון את מסד הנתונים ולחבר אליו את פרויקט NorthwindCS. אם לא, תוכל להצמיד את פרויקט NorthwindCS אל מסד הנתונים של Northwind במנהל מסד נתונים SQL Server.

פרויקט NorthwindCS מוסיף פריט **Show Me** (הראה לי) מיוחד בשורת התפריטים. אם תבחר בפריט זה, תוצג לפניך תיבת דו-שיח (הנראית בתרשים 12.2), המעמידה לרשותך את ההדגמה של פרויקט Access. במסגרת ההדגמה יוסברו התכונות המיוחדות של פרויקט Access וסוגי הנתונים השונים ב- SQL Server. על אף קיום התאמה מסוימת בין סוגי הנתונים ב-Jet וב- SQL Server, יש ביניהם גם הבדלים משמעותיים. לדוגמה, סוג הנתונים **Timestamp** הוא ייחודי ל- SQL Server. כמו כן, ב- SQL Server ניתן לייצג ערכי **Currency** (מטבע) וערכי **Date/Time** (תאריך/שעה) באופן שונה מאשר ב-Jet. נוסף לכך, ב- SQL Server קיימים במפורש סוגי נתונים Unicode וכאלה שאינם ב- Unicode.



תרשים 12.2: תיבת הדו-שיח **Show Me** בפרויקט NorthwindCS

הבדל עקרוני נוסף בין פרויקטים של Access וקבצי ‎*.mdb‎ מקובלים טמון בשימוש בתצוגות, בשגרות מאוחסנות ובשאילתות. תיבת הדו-שיח **Show Me** מסבירה כיצד התצוגות והשגרות המאוחסנות מחליפות את השאילתות ביישומים מותאמים אישית. זכור, שבעזרת קבצי ‎*.mdb‎, תצוגות ושגרות מאוחסנות הופכים לזמינים מול מנגנון מסד הנתונים Jet, אולם יש הבדל בין כללי התחביר החלים על תצוגות ב-Jet לעומת SQL Server. לדוגמה, קבצי ‎*.mdb‎ מאפשרים מיון רשומות בתצוגה באמצעות מונח ה-SQL ‏'ORDER BY'. המשפט ORDER BY ב-SQL עבור תצוגה אסור הן ב-SQL Server והן ב-MSDE. במנהלי מסדי נתונים אלה מסוג שרת-לקוח, משפט ORDER BY שמור לשגרות מאוחסנות בלבד.

אם תיבת הדו-שיח **Show Me** אינה עונה על צרכיך באופן מלא, נסה להיעזר בעזרה המקוונת של Access. פתח את העזרה המקוונת וחפש עזרה בנושא "עבודה עם פרויקט Microsoft Access". תמצא עזרה מקוונת נוספת בנושא פרויקטים של Access אם תפתח טבלה של פרויקט Access בתצוגת **עיצוב** (Design) ותלחץ על F1. סגור את חלון העזרה של Microsoft Access לפני הלחיצה על F1 כדי להיכנס לחלק אחר של מערכת העזרה. בתחום זה של העזרה המקוונת תמצא גם פרטים אודות Transact-SQL והודעות שגיאה ב-SQL Server. לבסוף, תוכל להסתייע גם ב-SQL Server Books Online (כדי להיעזר במשאב זה דרוש לך מחשב שמותקנת בו הגירסה המלאה של SQL Server 7).

שימוש במסד הנתונים Pubs עם פרויקטים של Access

אם עבדת בעבר עם ODBCDirect, או שפיתחת פתרון מותאם למסד נתונים מרוחק בעזרת Access, קרוב לוודאי כבר פגשת במסד הנתונים Pubs. תוכל להסתמך על היכרות זאת ולהרחיבה בעבודה עם פרויקטים של Access. זכור שדרוש לך עותק של SQL Server כדי לעבוד עם מסד הנתונים Pubs. תוכל להפעיל את מסד הנתונים עם פרויקט Access ב- SQL Server 7 או ב- SQL Server 6.5.

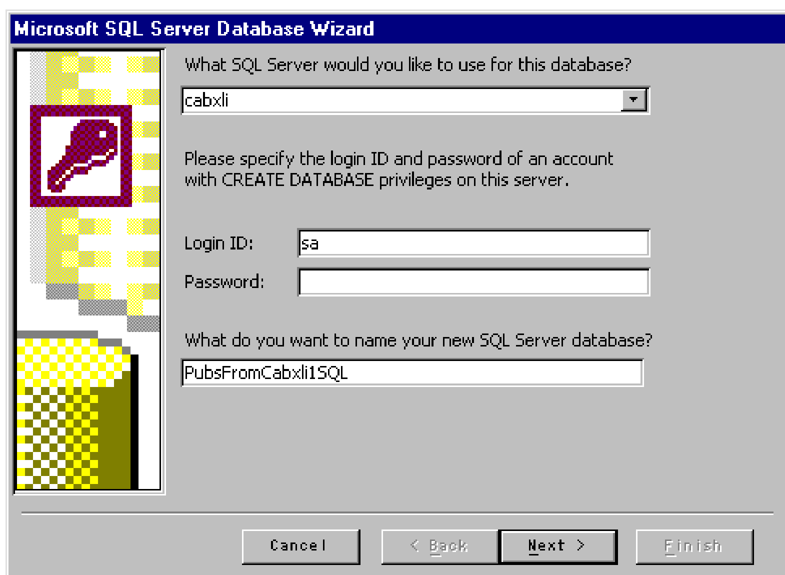
כדי לעבוד עם מסד הנתונים Pubs מתוך פרויקט Access דרוש לך קובץ adp.* שיצביע אל Pubs. אם לא עומד לרשותך קובץ adp.* מנותק, תוכל ליצור קובץ כזה על ידי יצירת פרויקט Access חדש ולאחר מכן שחרור מסד הנתונים שלו. בשלב הבא עליך לפתוח את תיבת הדו-שיח **Data Link Properties** ולכוון את פרויקט Access המנותק כך שיצביע אל מסד הנתונים Pubs.

כדי ליצור קובץ adp.* חדש, הפעל את Access וצור מסד נתונים חדש. לשם כך בחר בלחצן האפשרות **אשפי מסד נתונים, דפי ופרויקטים של Access** (Access Database Wizard, Page and Projects). פעולה זאת תפתח את תיבת הדו-שיח **חדש** (New) (לחילופין, תוכל ללחוץ על הפריט **חדש** בסרגל הכלים **מסד נתונים**). בחר בכרטיסיה **כללי** (General), ולאחר מכן לחץ לחיצה כפולה על הפריט **פרויקט (מסד נתונים חדש)** ((Project (New Database)). בתיבת הטקסט **שם הקובץ** (File Name) הקלד שם עבור קובץ adp.* החדש. לחיצה על **צור** (Create) מפעילה את **אשף מסד נתונים Microsoft SQL Server Database** (המוצג בתרשים 12.3). האשף מכיל את שם השרת המקומי (CABXLI בדוגמה שלנו), וכן הצעה של שם למסד הנתונים, המבוססת על שם הקובץ שהקלדת. עליך להזין את נתוני חשבון המשתמש (זיהוי משתמש וסיסמה – sa כזיהוי ותו ריק כסיסמה בדרך כלל מספיקים). כעת לחץ על **הבא** (Next) ולאחר מכן על **סיום** (Finish) כדי להשלים את התהליך. מרגע שפרויקט Access זמין, נתק אותו ממקור הנתונים שלו על ידי שחרור מסד הנתונים: בחר **כלים** (Tools), **עזרי מסד נתונים**(Database Accessories), **שחרור מסד נתונים של SQL** (Drop-SQL-Database). לבסוף, פתח את תיבת הדו-שיח **Data Link Properties** של פרויקט Access המנותק. הזן את הנתונים שיכוונו את קובץ הפרויקט אל מסד הנתונים Pubs ב- SQL Server.

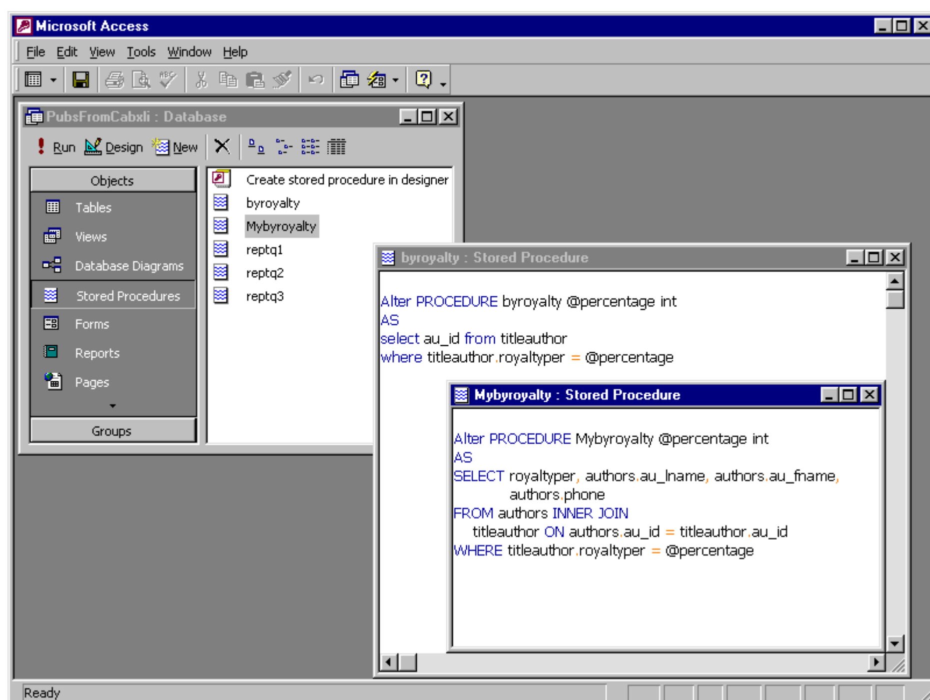


הערה:

אם כל שדרוש לך הוא קובץ adp.* מנותק (כדי לכוון אותו אל מסד נתונים Pubs או מסד נתונים אחר כלשהו), תוכל לקצר את התהליך. הקש על המקש Esc כאשר מופיע מסך הפתיחה של **אשף מסד נתונים Microsoft SQL Server Database**. לאחר מכן פתח את תיבת הדו-שיח **Data Link Properties** של קובץ adp.* המנותק שייווצר, וכוון אותו כך שיצביע אל מסד הנתונים הרצוי.



תרשים 12.3: מסך הפתיחה של אשף מסד נתונים **Microsoft SQL Server Database**



תרשים 12.4: השגרות המאוחסנות במסד הנתונים Pubs

לאחר שיצרת פרויקט Access המקושר אל מסד הנתונים Pubs, תוכל לשאוב ממנו דוגמאות מעניינות של הטכנולוגיה SQL Server שיאפשרו לך ללטש את הכלים שלך. תרשים 12.4 מציג את ארבע השגרות המאוחסנות ב-Pubs כאשר השיגרה byroyalty פתוחה. דוגמה זאת מחזירה את זיהויי המחבר של כל מחבר בעל הסכם על אחוזי תמלוגים, אשר תואם את הקלט היחיד בתגובה לשאילתת הפרמטרים. השיגרה המאוחסנת Mybyroyalty היא הרחבה של השיגרה הראשונה, והיא מחזירה את שמו הפרטי, שם המשפחה ומספר הטלפון של המחבר.

שחזור מסדי נתונים של SQL Server

SQL Server הוא מסד נתונים בעל עוצמה תעשייתית, אולם ניתן לפגוע במערכת באופן שגורם לאובדן היכולת לעבוד עם הנתונים. לדוגמה, כשל במדיה עלול להשחית את העותק הראשי של מסד הנתונים. כאשר הנזק חמור מאוד, ייתכן שלא תצליח כלל להפעיל את SQL Server. במקרה זה תיאלץ להסיר את ההתקנה של SQL Server או MSDE. להסרת התקנה פגומה, בחר באפשרות **Uninstall MSDE** ,**MSDE** מתפריט **התחלה** (Start) או מלוח הבקרה. פעולה זאת תסיר את כל מסדי הנתונים של המערכת, כמו העותק הראשי של מסד הנתונים, אולם תשאיר על כנם מסדי נתונים של משתמשים, כמו למשל NorthwindCS.

כאשר תתחיל לעבוד עם MSDE או עם SQL Server, בוודאי תחזור שוב ושוב על התקנת מנהל מסד הנתונים שרת-לקוח כדי להעמיק את היכרותך עם התהליך. לאחר התקנה מחדש של SQL Server או MSDE, העתק אליהם את העותק הראשי החדש מתוך עותק גיבוי עדכני של העותק הראשי של מסד הנתונים, או בנה מחדש את העותק הראשי כך שיכיר במסדי הנתונים הקיימים. כל אחת משתי השיטות הללו תאפשר לך להתחבר עם מסדי הנתונים הקיימים.

רצוי בהחלט לשמור עותק שוטף של העותק הראשי של מסד הנתונים, הואיל ודבר זה עשוי לפשט את פעולת החיבור של התקנה חדשה של SQL Server עם מסדי נתונים קיימים. צור גיבוי לעותק הראשי של מסד הנתונים לאחר יצירה או מחיקה של מסד נתונים חדש של משתמש, או חשבונות כניסה למערכת. הוספת משתמש חדש למסד נתונים אינה משנה את העותק הראשי, מכיון שנתוני אבטחה, כמו חשבונות משתמשים, נכנסים אל תוך קובץ מסד הנתונים.

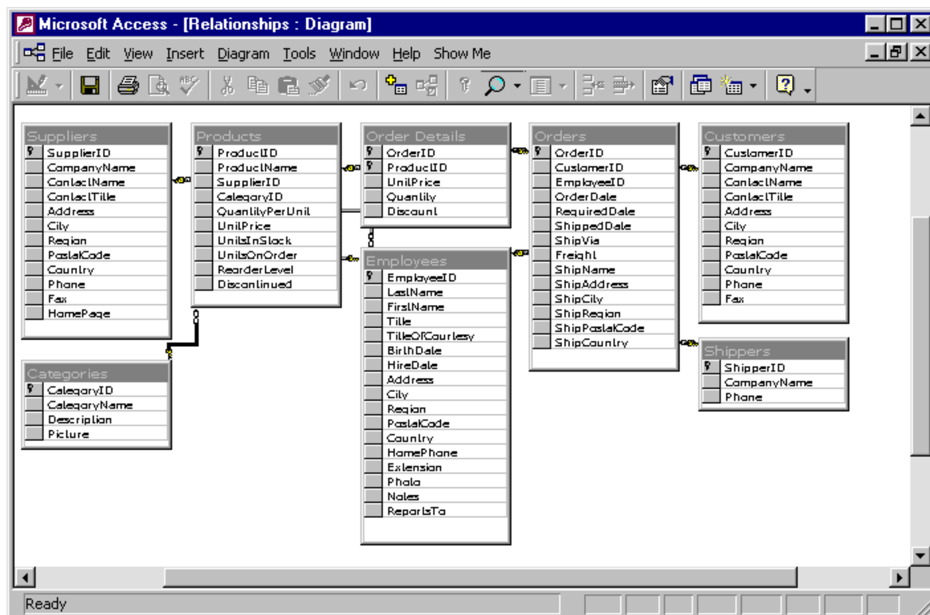
שחזור העותק הראשי כרוך בשינוי השם של קבצי מסדי הנתונים הישנים של המשתמשים, ויצירת מסדי נתונים של מצייני מיקום עם שמות מסדי הנתונים הקודמים. עליך ליצור פרויקט Access עבור כל קובץ מסד נתונים שבכוונתך לשחזר. כאשר אתה יוצר פרויקט Access חדש, ערוך את השם בתיבות הדו-שיח שלו באשף מסדי הנתונים Microsoft SQL Server Database, והקצה לו את שמו של אחד מקבצי מסד הנתונים שאתה מבקש לכלול בעותק הראשי החדש. לאחר שבנית פרויקט Access חדש עבור כל אחד מקבצי מסדי הנתונים שאתה מבקש לשחזר, העתק את קבצי מסדי הנתונים הישנים (אלה שאת שמם שינית) על גבי הקבצים החדשים שיצרת. פעולה זאת מעדכנת את העותק הראשי כך שיזהה את קבצי מסדי הנתונים של המשתמשים מההתקנה הקודמת. כל קובץ ישן מועתק אל שם שהעותק הראשי מכיר.

דיאגרמות וטבלאות של מסדי נתונים

דיאגרמה של מסד נתונים היא אוסף של אובייקטי מסד נתונים, בדומה לטבלאות, תצוגות ושגרות מאוחסנות. Access 2000 שומר אותן יחד עם קובץ מסד הנתונים של SQL Server או MSDE, שלא כמו פרויקט Access. בעוד שדיאגרמה של מסד נתונים נפרד עשויה לדמות לתצוגה המופיעה בחלון **קשרי גומלין** (Relationships) של קובץ ‎*.mdb‏ קלאסי ב-Access, דיאגרמות של מסדי נתונים שרת-לקוח גמישות הרבה יותר, ומאפשרות שליטה הדוקה יותר בעיצוב מסד הנתונים. לדוגמה, תוכל ליזום עיצוב של טבלאות חדשות מתוך חלון של דיאגרמת מסד נתונים.

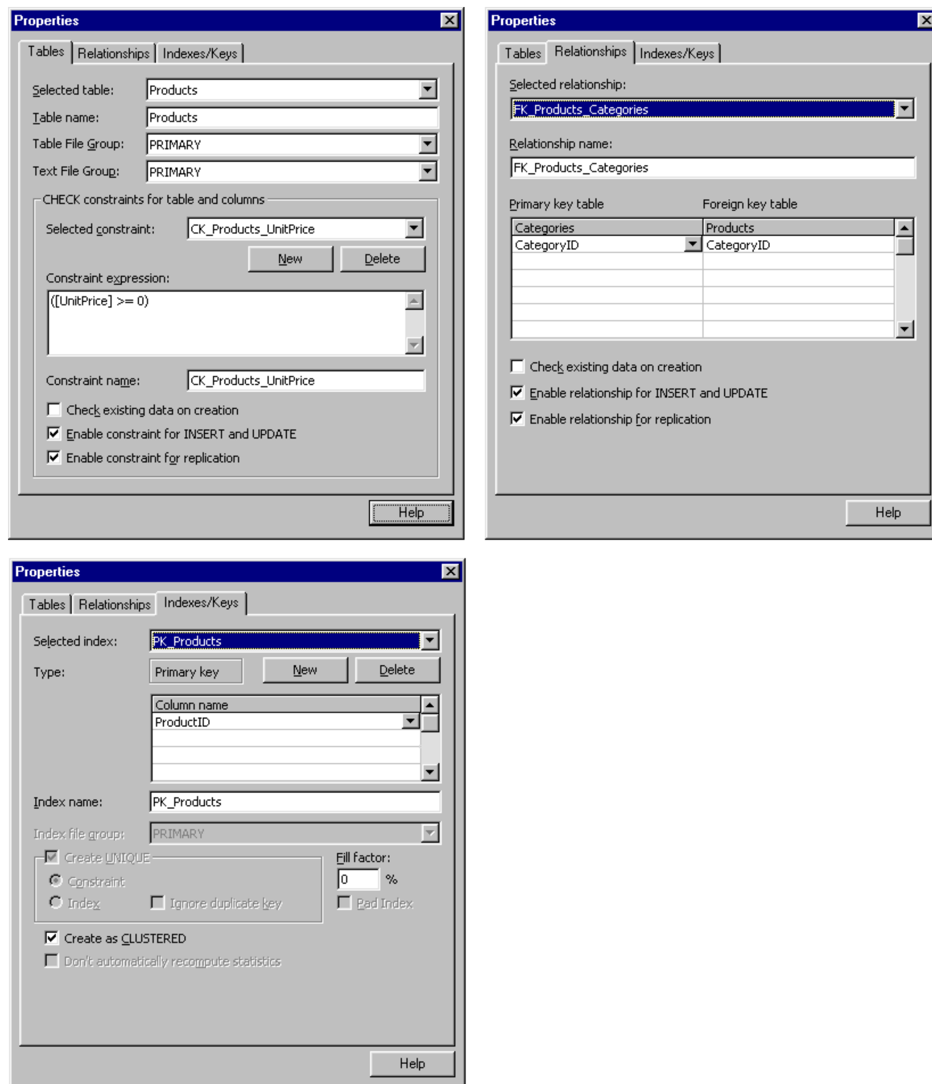
מיפוי קשרי גומלין באמצעות דיאגרמות של מסדי נתונים

תרשים 12.5 מציג את דיאגרמת מסד הנתונים של מסד הנתונים NorthwindCS. בדומה לחלון **קשרי גומלין** בקבצי Access רגילים למסדי נתונים Jet, הדיאגרמה מתארת את הטבלאות שבעיצוב מסד הנתונים ואת קשרי הגומלין ביניהן. תוכל לבצע כיוונון עדין של קשרי הגומלין בין שתי טבלאות על ידי לחיצה על הקו המחבר ביניהן, ובחירה באפשרות **מאפיינים** (Properties) מתפריט הקיצור. בתרשים 12.5 מוצג קשר הגומלין בין הטבלאות Products ו-Categories, כשהוא נבחר.



תרשים 12.5: דיאגרמת מסד הנתונים מתוך פרויקט NorthwindCS

תרשים 12.6 מציג את תיבת הדו-שיח **Properties** (מאפיינים) עבור קשר הגומלין בין שתי הטבלאות שנבחר בתרשים 12.5. הכרטיסיות מכילות הרבה יותר מידע אודות קשרי הגומלין בין הטבלאות מאשר התצוגה בחלון **קשרי גומלין** בקובץ מסד נתונים קלאסי של Jet או Access.



תרשים 12.6: תיבת הדו-שיח Properties עבור קשר הגומלין בין Products ל- Categories במסד הנתונים NorthwindCS

השתמש בכרטיסיה **Tables** (טבלאות) כדי לשלוט בתנאי האימות שיחולו על נתונים המוזנים אל תוך הטבלאות. למשימה זאת, השתמש במגבלות. כדי לבחור הגבלה, לערוך אותה או למחוק אותה, בחר בהגבלה הרצויה מתוך תיבת הרשימה הנפתחת

Selected Constraint (מגבלה נבחרת). בחר באפשרות **New** (חדש) כדי להחיל הגבלה חדשה על טבלה. לחץ על **Delete** (מחק) כדי למחוק את ההגבלה הנבחרת באותו רגע. שלוש תיבות הסימון שבתחתית הכרטיסיה **Tables** קובעות מתי הגבלה מסוימת תיכנס לתוקפה. שתי הבחירות בפינה השמאלית העליונה בתרשים 12.6 מחילות את ההגבלה כאשר משתמש מוסיף או מעדכן נתונים בטבלה, או כאשר הטבלה מועתקת אל מסד נתונים אחר. השתמש ברשימה הנפתחת **Selected Table** (טבלה נבחרת) בראש הכרטיסיה **Tables** כדי לבחור טבלה אחרת ולהחיל עליה הגבלות נוספות.

במצב הראשוני הכרטיסיה **Relationships** (קשרי גומלין) מראה את קשר הגומלין בין הטבלאות Products ו- Categories. באפשרותך לבחון גם את קשרי הגומלין של הטבלה Products עם טבלאות אחרות, כמו Suppliers או Order Details, וזאת באמצעות תיבת הרשימה הנפתחת **Selected Relationships** (קשרי גומלין נבחרים). כדי להגיע לתוצאה זהה בקובץ mdb.* עליך לבחור במפורש את קשר הגומלין הרצוי בחלון **קשרי גומלין**.

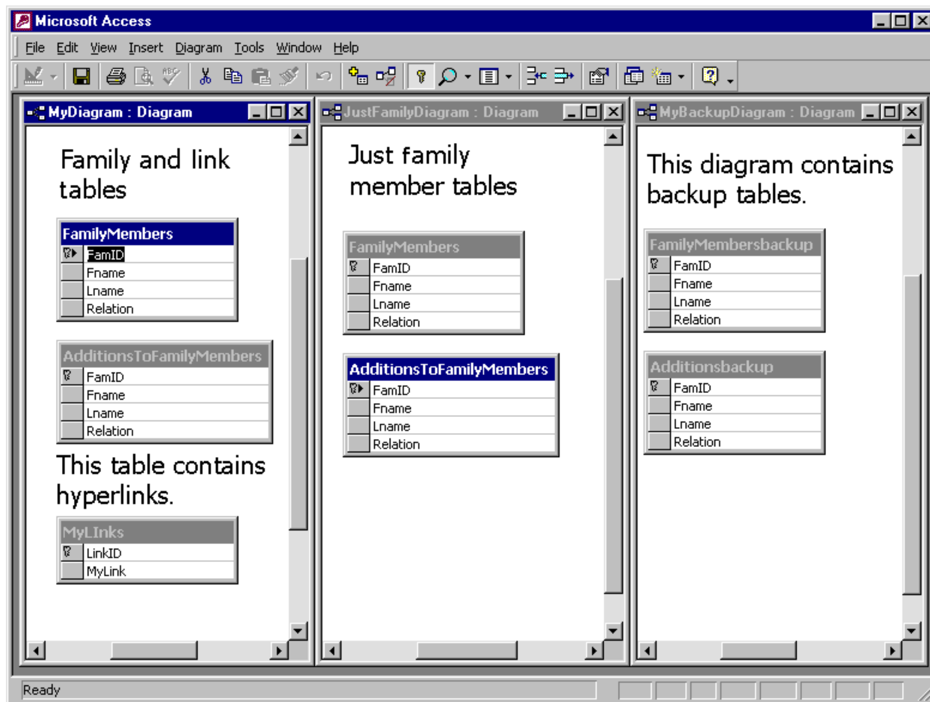
הכרטיסיה **Indexes/Keys** (אינדקסים/מפתחות) מציגה הגדרות עבור המפתחות והאינדקסים הקיימים של הטבלה, ומאפשרת הוספת פריטים חדשים או מחיקת פריטים קיימים. השדה ProductID מגדיר מפתח ראשי עבור טבלה Products. האינדקס מאורגן באשכולות, ופירוש הדבר זהות בין הסדר הפיסי והסדר הלוגי של השורות.

ניהול דיאגרמות של מסדי נתונים מרובים

תרשים 12.7 מציג שימוש אפשרי נוסף בדיאגרמות של מסדי נתונים. בתרשים נראות שלוש דיאגרמות שונות, שכולן מתייחסות אל אותו מסד נתונים. קבצי mdb.* קלאסיים אינם מאפשרים הצגה של דיאגרמות מרובות המתארות את קשרי הגומלין בתוך מסד נתונים; לכל קובץ חלון **קשרי גומלין** יחיד. לעומת זאת, ב- SQL Server ניתן להציג חלונות מרובים למסדי נתונים. חלונות של דיאגרמות מסדי נתונים עשויים להכיל תכנים בלעדיים או חופפים. לדוגמה, שתי הטבלאות – FamilyMembers ו-AdditionsToFamilyMembers – מופיעות הן בחלון MyDiagram והן בחלון JustFamilyDiagram. תוכל להעתיק חלון אל תוך הלוח, ולהדביק אותו בחלון אחר. הואיל וניתן לאבטח חלונות של דיגארמות מסדי נתונים, תוכל לשלוט בתצוגות מסד הנתונים שיעמדו לרשות קבוצות שונות של משתמשים.

כפי שניתן לראות בתרשים 12.7, אפשר להוסיף תוויות לדיאגרמות כדי לזהות את אחת הטבלאות, או את כולן. כדי לשנות סגנון, גודל, צבע או מאפיין אחר של גופן בתווית, לחץ לחיצה ימנית על תיבת תווית כדי לפתוח את תיבת הדו-שיח **גופן** (Font).

תוכל לערוך את רשימת החלונות של דיאגרמות מסד נתונים בתוך מסד נתונים מסוים על ידי ספירת חברי האוסף AllDatabaseDiagrams. כמו כן התוכניות שלך תוכלנה לקרוא את המאפיין IsLoaded של חברי האוסף ולדווח אם חלון מסוים פתוח או לא. עם זאת, לא ניתן להגדיר את המאפיין IsLoaded כך שיפתח ויסגור חלונות של דיאגרמות מסד נתונים. האוסף AllDatabaseDiagrams שייך לאחד משני האובייקטים – CurrentData או CodeData – בתוך היישום.



תרשים 12.7: שלושה חלונות של דיאגרמות עבור אותו מסד נתונים

השיגרה הבאה יוצרת את רשימת כל החלונות של דיאגרמות מסד נתונים, ומציינת ליד כל חלון אם הוא פתוח, או לא:

```
Sub listDatabaseDiagrams()
Dim obj As AccessObject, dbs As Object

' Set the hierarchical container for AllDatabaseDiagrams.
    Set dbs = Application.CurrentData

' List database diagrams by whether they are loaded.
    For Each obj In dbs.AllDatabaseDiagrams
        If obj.IsLoaded = True Then
            Debug.Print obj.Name; " is loaded."
        Else
            Debug.Print obj.Name & " is not loaded."
        End If
    Next obj
End Sub
```

עריכת טבלה בחלון של דיאגרמת מסד נתונים

באפשרותך לערוך טבלאות או להוסיף טבלאות חדשות דרך חלון של דיאגרמת מסד נתונים או באמצעות האוסף **אובייקטים** (Objects), **טבלאות** (Tables) שבחלון **מסד נתונים** (Database). לכל גישה יתרונות משלה לביצוע משימות שונות.

חלונות של דיאגרמות מסדי נתונים שימושיים ביותר לביצוע משימות של הגדרת נתונים באופן גרפי או בתיבת הדו-שיח **מאפיינים** (Properties) של טבלה נתונה. כדי להקצות מפתח ראשי, בחר בבורר השורות של השדה בתוך דיאגרמת מסד הנתונים. אם המפתח הראשי מבוסס על יותר משדה אחד, לחץ לחיצה רצופה על מקש Ctrl ובחר את שאר השדות המשתתפים בהגדרת המפתח הראשי. להשלמת המשימה, לחץ לחיצה ימנית על השורה או השורות שנבחרו, ובחר באפשרות **מפתח ראשי** (Primary Key) מתפריט הקיצור.

חלונות של דיאגרמות מסד נתונים שימושיים גם להגדרת קשרי גומלין. התחל בבחירת בורר השורות עבור המפתח הראשי בטבלה הראשונה. אינך חייב להשתמש במפתח ראשי, אולם חובה להציב בשדה ערכים ייחודיים. כעת, גרור את השדה הנבחר מהטבלה הראשונה ושחרר אותו בפס הכותרת של הטבלה השנייה. תיבת הדו-שיח **Create Relationship** (שדומה דמיון רב לחלונית הימנית העליונה בתרשים 12.6) תופיע על המסך. שנה את קשרי הגומלין כנדרש, ולחץ על **OK**.

תוכל לכפות שלמות קשרים באמצעות שלוש תיבות הסימון שבתחתית תיבת הדו-שיח **Create Relationship** ובכרטיסיה **Relationship** שבתיבת הדו-שיח **Properties** (הנראית בתרשים 12.6). בחר בתיבת הסימון **Enable Relationship For Insert And Update** כדי לכפות שלמות קשרים על כל הנתונים החדשים במסד הנתונים הנוכחי. אם שלמות הקשרים כבר חלה על הנתונים הקיימים, או אם אין צורך בכך, הימנע מבחירת תיבת הסימון **Check Existing Data On Creation**. אם בשל החלת שלמות קשרים במסד נתונים האמור לקבל שורות ממסד הנתונים שלך, ההעברה של רשומות רבות תיכשל (בגלל הפרת כללי מפתחות), השאר את תיבת הסימון **Enable Relationship For Replication** ללא בחירה.

עריכה ויצירה של טבלאות באמצעות גיליון עבודה

תוכל להוסיף למסד נתונים טבלה חדשה על ידי לחיצה ימנית בחלון של דיאגרמת מסד נתונים ובחירה באפשרות **טבלה חדשה** (New Table). לאחר שקיבלת את השם שהפיקה המערכת או שהקצית שם חדש, יופיע על המסך גיליון עבודה. גיליון עבודה זה משמש למתן שמות לשדות, להקצאת סוגי נתונים ולפירוט הגדרות נתונים נוספות. גיליון עבודה זהה נפתח מתוך האוסף **טבלאות** (Tables) שבחלון **מסד נתונים** כאשר לוחצים על האפשרות **חדש** (New) או **עיצוב** (Design). גיליון העבודה חושף מספר מאפייני טבלה ושדה ידידותיים למשתמש.

תרשים 12.8 מציג שני גליונות עבודה עבור הטבלאות FamilyMembers ו-MyLInks שהופיעו גם בתרשים 12.7. למעשה, קשרי הגומלין עודכנו בהשוואה לתרשים 12.7, כך שמתקיימים בין הטבלאות קשרי גומלין מסוג יחיד לרבים. השדה FamID בטבלה MyLInks קשור אל השדה הראשי FamID בטבלה FamilyMembers. תוכל ליצור באחת הטבלאות מפתח זר שיהיה תואם בדיוק למפתח הראשי בטבלה אחרת, ממש כפי שנהוג בקובץ *.mdb הקלאסי ב-Access. העתק את המפתח הראשי מהטבלה הראשונה אל הלוח, והדבק אותו בטבלה השנייה.

MyLInks : Table

Column Name	Datatype	Length	Precision	Scale	Allow Nulls	Default Value	Identity	Identity Seed	Identity Increment	Is RowGuid
LinkID	int	4	10	0			✓	1	1	
MyLink	char	125	0	0	✓					
FamID	int	4	10	0	✓					

FamilyMembers : Table

Column Name	Datatype	Length	Precision	Scale	Allow Nulls	Default Value	Identity	Identity Seed	Identity Increment	Is RowGuid
FamID	int	4	10	0			✓	1	1	
Fname	char	20	0	0						
Lname	char	25	0	0						
Relation	char	30	0	0						

תרשים 12.8: שני גליונות עבודה המראים את האפשרויות להגדרת שדות בטבלאות של מסדי נתונים MSDE או SQL Server

המפתח הראשי FamID בטבלה FamilyMembers מכיל נתונים מסוג Integer של SQL Server. סוג נתונים זה מקביל לסוג הנתונים Long במסדי נתונים של Jet או Access. המפתח הראשי LinkID בטבלה MyLInk משתמש בסוג נתונים Integer עם הגדרת עמודה Identity. שים לב, שממשק זה, שלא כמו ב- Access עם Jet, מכיל ממשק גרפי להגדרת ערכי זיהוי תחיליים וערכי צעד. סוג הנתונים nvarchar מיוחד לשדות המחרוזת באורך משתנה ב-Unicode. סוג הנתונים המקביל לנתונים שאינם ב- Unicode הוא varchar. גם במקרה זה, Access עם Jet אינו חושף את ההבחנה בין סוגי הנתונים.

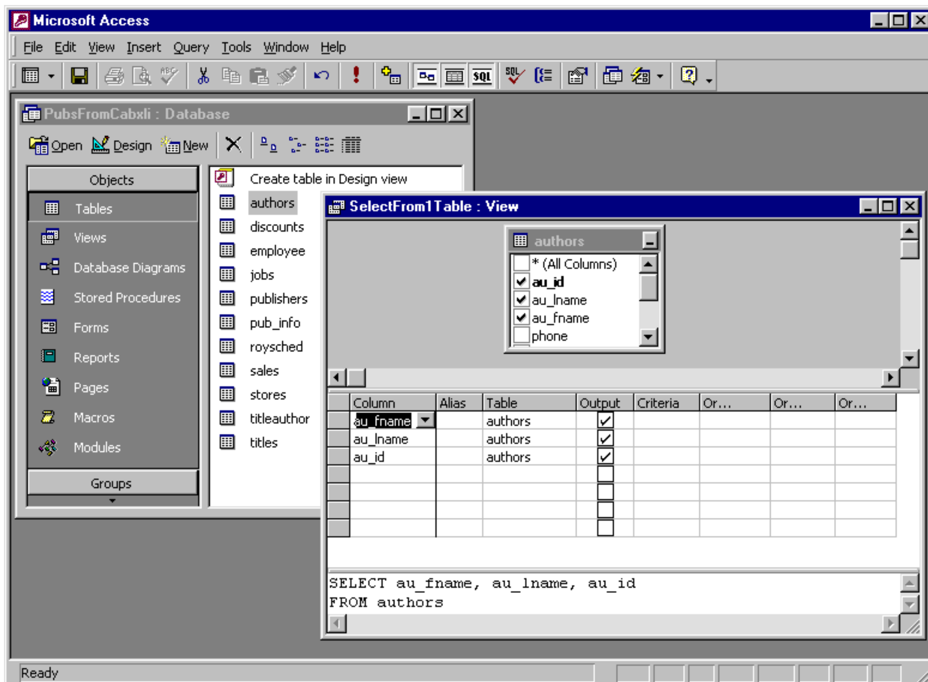
תצוגות ושגרות מאוחסנות

שלא כמו Access, SQL Server אינו מכיל שאילתות לבניית יישומים מותאמים אישית, אולם יש בו רכיבים פונקציונליים דומים, המושגים באמצעות תצוגות ושגרות מאוחסנות. תצוגה היא משפט SQL המחזיר שורות מבלי להשתמש בפרמטרים או במשפט ORDER BY. לעומת זאת שיגרה מאוחסנת יכולה להשתמש הן בפרמטרים והן במשפט ORDER BY כדי לציין קבוצת שורות להחזרה. שגרות מאוחסנות יכולות להוציא לפועל גם משפטי SQL שאינם מחזירים שורות, כמו INSERT, UPDATE ו- DELETE.

שימוש בתצוגות

כדי להקל עליך את ההסתגלות לשינוי בכללי כתיבת השאילתות במעבר מ-Jet ל-MSDE או ל-SQL Server, הסתייע במעצב השאילתות החדש של Access, הדומה דמיון רב למעצב השאילתות לשאילתות המבוססות על Jet. השתמש במעצב השאילתות לבניית תצוגות (את השגרות המאוחסנות תיאלץ לכתוב ללא עזרתו של אשף בנייה). מעצב השאילתות בנוי משלושה מסכים, שניתן להפעיל או לכבות. בניית התצוגה יכולה להיעשות בכל אחד משלושת המסכים, וכל ערך המוזן בלוח אחד מעדכן אוטומטית גם את שני הלוחות האחרים.

תרשים 12.9 מציג את מעצב השאילתות בבנייה של שאילתת בחירה המתייחסת אל הטבלה authors שבמסד הנתונים Pubs. המסך העליון הוא חלונית הדיאגרמה. לחץ לחיצה ימנית על חלונית זאת כדי להוסיף כל טבלה או תצוגה קיימת כמקור קלט לתצוגה החדשה. בחר את תיבת הסימון הסמוכה לשדה שאתה מבקש להוסיף לחלונית הרשת שבמרכז. השתמש בתיבות הסימון בעמודת הקלט כדי לציין אם שדה מסוים אמור להופיע בקבוצה המוחזרת של תצוגה נתונה. כמו כן תוכל להוסיף קריטריונים בעמודות שמימין כדי להגביל את מספר השורות שתצוגה יכולה להחזיר. אם אתה מעדיף כתיבת SQL על פני הצבעה ולחיצה, השתמש בחלונית SQL (החלונית התחתונה) לכתיבת משפטי SQL.



תרשים 12.9: מעצב השאילתות

בסרגל הכלים **הצג עיצוב** (View Design) ובשורת התפריטים תמצא עזרה נוספת לכתיבת תצוגות. הלחצן SQL מפעיל ומשבית את חלונית ה- SQL. לפי ברירת המחדל, הלחצן שמשמאל ללחצן SQL מבצע פונקציה דומה עבור חלונית הרשת וחלונית הדיאגרמה. הלחצן שמימין, המסומן בסימן ✓ (לחצן **בדוק תחביר SQL** (Verify SQL)) מבצע בדיקת תחביר למשפטי SQLשאתה מזין היישר אל תוך חלונית ה- SQL. הלחצן **קיבוץ לפי** (Group By) הסמוך ללחצן **בדוק תחביר SQL** משנה את חלונית הרשת כדי להתחיל בעיצוב של שאילתת צבירה. התנהגותו דומה לזו של הלחצן **Totals** (Σ) בבונה השאילתות הקלאסי של Jet. לאחר שעיצבת את התצוגה כרצונך, לחץ על הלחצן **אובייקט חדש** כדי לבצע אוטומציה של הכנת טופס או דוח המבוססים על התצוגה שעיצבת (אם טרם שמרת את התצוגה, תתבקש לשמור אותה תחילה).



הערה:

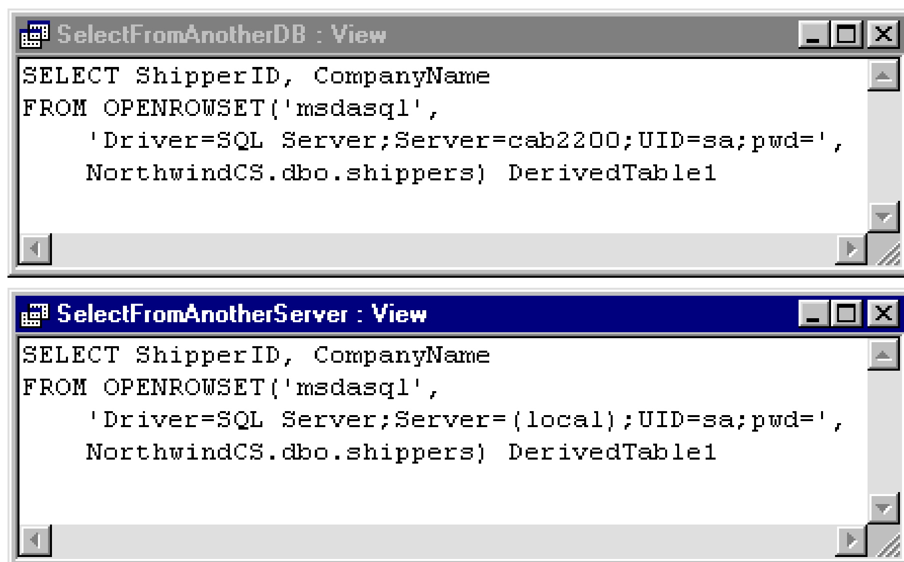
מעצב השאילתות כולל את היתרונות של עיצוב שאילתות גרפיות לבניית תצוגות ללא הטרחה הנוספת של הצמדת טבלאות מרוחקות (כנדרש עם קבצי mdb.* קלאסיים). משפטי SQL עבור התצוגות מתבצעים בשרת מסד הנתונים, ואילו פרויקט Access רק מציג את הקבוצה המוחזרת.

אינך מוגבל לבניית תצוגות השואבות נתונים מהחיבור הפעיל של פרויקט Access. באפשרותך להשתמש במילת המפתח OPENROWSET כדי לכוון תצוגה אל מסד נתונים אחר באותו שרת, או אפילו בשרת אחר. דבר זה יפה גם לשרתים מסוגים שונים, כמו למשל Oracle או Jet. מילת המפתח OPENROWSET מקבלת שלושה ארגומנטים מופרדים זה מזה בפסיקים. הוסף גרש בתחילת ובסוף הארגומנט הראשון והשני כדי לציין שמדובר במחרוזות. הארגומנט השלישי הוא שם של אובייקט מסד נתונים. הארגומנט הראשון מכיל את שם ספק OLE DB עבור מקור הנתונים החלופי.

בשתי הדוגמאות הבאות נעשה שימוש בספק OLE DB עבור מקורות נתונים תואמי-ODBC. הארגומנט השני מכיל את מחרוזת החיבור. הספק קובע את התחביר הנכון לארגומנט זה. הארגומנט השלישי הוא אובייקט מסד נתונים בתבנית SQL Server. התבנית הכללית לציון האובייקט היא linkserver.catalog.schema.object. linkserver הוא השם המקומי של SQL Server 7 עבור ספק ה- OLE DB שמצביע אל מקור הנתונים ההטרוגני המרוחק. אם אינך מוציא לפועל שאילתה עבור מקור נתונים שאינו SQL Server, אינך זקוק לפרמטר זה. catalog הוא שם מסד הנתונים, ו- schema מציין את הבעלים של מסד הנתונים. Object הוא שם הטבלה.

תרשים 12.10 מראה ייצוגים של שתי תצוגות בחלוניות SQL. שניהם לקוחים מקובץ הפרויקט PubsFromCabxli.adp. לצורך סביבת הבדיקות, קובץ זה שוכן במחשב ששמו CABXLI, אולם בתיבת הדו-שיח שלו **Data Link Properties** מצוין קשר עם מסד הנתונים Pubs השוכן במחשב ששמו cab2200. כל תצוגה משתמשת במילת המפתח OPENROWSET כדי להתחבר אל מקור נתונים שאינו מסד הנתונים Pubs במחשב cab2200. התצוגה SelectFromAnotherDB מתחברת את הטבלה Shippers

שבמסד הנתונים NorthwindCS, במחשב cab2200. התצוגה SelectFromAnotherServer מתחברת אל הטבלה Shippers בשרת המקומי – CABXLI. מעצב השאילתות מוסיף באופן אוטומטי את המונח DerivedTable1 בסופם של שני משפטי SQL.



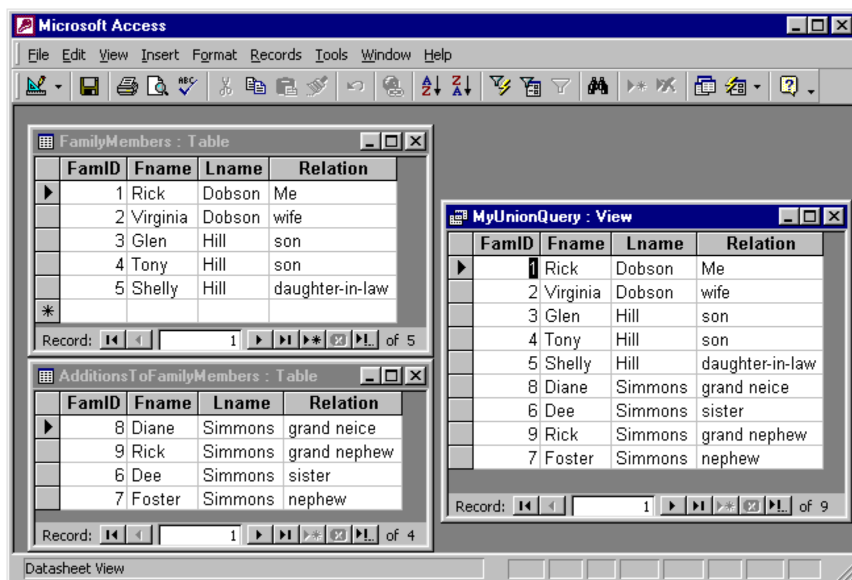
תרשים 12.10: תחביר SQL עבור מילת המפתח OPENROWSET, המשמשת ליצירת חיבור עם מקורות נתונים שמחוץ לחיבור הפעיל.

שימוש בשגרות מאוחסנות

באפשרותך להשתמש בשגרות מאוחסנות לבנייה נוספת והגדלת העוצמה של תצוגות ביישומי SQL Server מותאמים אישית. לא קיים מעצב שאילתות לשגרות מאוחסנות. טכניקת השגרות המאוחסנות מפצה על חיסרון זה על ידי שימוש באוצר מילים משופר המבוסס על Transact-SQL, שהיא גירסה מיוחדת של שפת SQL עבור SQL Server. הגמישות של Transact-SQL בשילוב עם העוצמה של השגרות המאוחסנות שלה מעניקות לגירסה זו דירוג שני או שלישי כשפת תכנות מועדפת, אחרי VBA ו- ADO, וזאת במיוחד אם בכוונתך לפתח בעזרת פרויקטים של Access.

תרשים 12.11 מציג את התוצאה של שאילתת איחוד בפרויקט של Access. זכור, ששאילתת איחוד משרשרת קבוצות של רשומות בזו אחר זו. במקרה זה, הקוד (המובא להלן) משרשר את הטבלה FamilyMembers עם הטבלה AdditionsToFamilyMembers. הקוד והנתונים לדוגמה עבור שאילתת האיחוד נמצאים בקובץ adp1.adp, בתקליטור המצורף. הטבלה AdditionsToFamilyMembers מפנה אל מסד הנתונים adp1SQL.

```
SELECT FamID, Fname, Lname, Relation
FROM AdditionsToFamilyMembers
UNION
SELECT FamID, Fname, Lname, Relation
FROM FamilyMembers
```



תרשים 12.11: שאילתת איחוד ב- SQL Server (וב- MSDE) יכולה לשרשר שתי טבלאות, או יותר, אולם אינה יכולה למיין אותן

הקוד נמצא בתצוגה ששמה MyUnionQuery. שים לב, שהשורות אינן ממוינות לפי השדה FamID מכיוון שאין ב- SQL משפט ORDER BY עבור תצוגה זאת. זכור שתצוגות ב- SQL Server אינן מאפשרות שימוש במשפט ORDER BY. אם ברצונך למיין את הרשומות לפי FamID או לפי כל שדה אחר, עליך להפעיל לשם כך שיגרה מאוחסנת.

הדוגמה הבאה מתאימה את ה- SQL לתצוגה ובונה שיגרה מאוחסנת גמישה יותר. במקרה זה, השיגרה ששמה StoredProcedure1 מבצעת את השרשור וממיינת לפי FamID.

```
ALTER PROCEDURE StoredProcedure1 AS
SELECT FamID, Fname, Lname, Relation
FROM AdditionsToFamilyMembers
UNION
SELECT FamID, Fname, Lname, Relation
FROM FamilyMembers
-- Here's the ORDER BY phrase
ORDER BY FamID
```

שמו של המשפט ALTER PROCEDURE נגזר מיכולתו לשנות שיגרה שנוצרה לראשונה בעזרת המשפט CREATE PROCEDURE. עם זאת, ניתן ליצור שיגרה מתחילתה בעזרת ALTER PROCEDURE. שים לב, שהתחביר מחייב להוסיף את מילת המפתח AS לאחר המשפט ALTER PROCEDURE. יתרת השיגרה המאוחסנת זהה לתצוגה MyUnionQuery, להוציא שתי השורות האחרונות. השורה הלפני-אחרונה מראה את התחביר להערה שנדרשת לה שורה אחת בלבד. התחל את השורה בשני תווי מקף (**--**). השורה האחרונה מכילה את משפט ORDER BY הממיין את הרשומות לפי FamID.

כאשר משתמשים בפרויקטים של Access, רצוי לרכוש בקיאות בשגרות מאוחסנות מטעמים רבים. מנגנוני SQL Server ו- MSDE תומכים בסדרה של פונקציות מינהליות באמצעות שגרות מאוחסנות של המערכת. שגרות אלה שוכנות במסד הנתונים הראשי. כל שגרות המערכת הללו נפתחות ב- _sp. מסיבה זאת, עליך להימנע מהשימוש בצירוף _sp בשמות השגרות המאוחסנות המותאמות שלך.

שיגרה מאוחסנת בשם sp_server_info מספקת משוב אודות כל סוגי הפונקציונליות בשרת (SQL Server Books Online מכיל מידע אודות שיגרה זאת). תוכל להפעיל אותה ביישום שלך כשיגרה מאוחסנת כדי לקבוע למנגנון SQL Server הגדרות מאפיינים שיסייעו לך להחליט כיצד לתכנת אותו. לדוגמה, השורה ה-18 של קבוצה מוחזרת רגילה עשויה לציין אם השרת ימיין תוך התחשבות באותיות רישיות, אם לאו. הדוגמה הבאה בונה ביישום שלך שיגרה מאוחסנת פשוטה המפעילה את שגרת המערכת המאוחסנת sp_server_info כדי לברר את כללי האיסוף בשרת שאתה מחובר אליו כרגע. הדוגמה מראה גם את התבנית להערה המשתרעת על פני יותר משורה אחת.

```
ALTER PROCEDURE DetermineSortOrder
/*If row 18 shows sort_order - nocase,
then sorts are case insensitive.
*/
AS
EXEC sp_server_info 18
```

סיבה נוספת להרחבת הידע אודות שגרות מאוחסנות קשורה בניצול הפרמטרים שלהן. תוכל להעביר אליהן פרמטרים כדי לקבל ערכי החזרה באמצעות פרמטרי הפלט. השיגרה DtermineSortOrder שולפת את השורה ה-18 מתוך הקבוצה המוחזרת שהפיקה השיגרה המאוחסנת sp_server_info. המספר 18 הוא פרמטר המציין לשגרת המערכת איזו שורה עליה להציג מתוך הקבוצה המוחזרת הרגילה שלה.

שתי השגרות המותאמות אישית הבאות מעצבות שגרות מאוחסנות שמקבלות פרמטרים, ומעבירות אליהן ערכים באמצעות קוד. שגרות מאוחסנות מותאמות אישית עם פרמטרים יכולות להציג באופן אוטומטי דרישה לערכים אם אינך מעביר להן ערכים באמצעות קוד. השיגרה הראשונה מבצעת שתי פעולות: היא מפעילה את השיגרה שמפיקה דוח מכירות, ולאחר מכן מעבירה ערכים אל השיגרה השנייה, שקובעת לדוח תאריכי תחילה וסיום, וכן שם של עובד. תוכל לערוך את שלושת הפרמטרים בשיגרה הראשונה כדי לשנות את הפלט של השיגרה השנייה.

```
ALTER PROCEDURE RunEmployeeSalesByDate
AS
EXEC "Employee Sales by Date" '1/1/95', '1/1/98', 'Buchanan'
```

השיגרה השנייה, לעיל, מקבלת את הפרמטרים כחלק מהמשפט ALTER PROCEDURE. כל ציון פרמטר מכיל שני ארגומנטים: השם, כמו למשל ‎@Beginning_Date, וכן סוג נתונים, כמו למשל datetime או varchar(20). על ציון סוג הנתונים המתייחס לשם של עובד בשיגרה המאוחסנת להתאים להגדרת סוג הנתונים עבור **שם העובד** בטבלה Employees. משפט SQL מחבר שלוש טבלאות, ופסוקית WHERE מפעילה את הפרמטרים.

```
/*This procedure accepts three parameters to determine the
content for a sales report.
    @Beginning_Date is the start date period.
    @Ending_Date is the end period.
    @lastname specifies the employee for which
    the report lists sales.
*/
ALTER PROCEDURE [Employee Sales by Date]
@Beginning_Date datetime, @Ending_Date datetime,
@lastname varchar(20)
AS
SELECT Employees.Country, Employees.LastName, Employees.FirstName,
    Orders.ShippedDate, Orders.OrderID,
    "Order Subtotals".Subtotal AS SaleAmount
FROM Employees INNER JOIN (Orders INNER JOIN "Order Subtotals" ON
    Orders.OrderID = "Order Subtotals".OrderID)
    ON Employees.EmployeeID = Orders.EmployeeID
-- This format is for comments on a single line.
WHERE Orders.ShippedDate Between @Beginning_Date And @Ending_Date And
    Employees.LastName = @lastname
```

דוחות וטפסים

לאחר שהתחברת אל מקור נתונים מרוחק אחד, או יותר, ולאחר שביצעת סינון, צבירה או שילוב בינם לבין מקורות אחרים, תוכל להציג בעזרת Access 2000 את התוצאות שקיבלת. הואיל וקיים שילוב הדוק בין טבלאות, תצוגות ושגרות מאוחסנות לבין טפסים, דוחות ומודולים, תוכל להגיש בנקל נתוני שרת-לקוח בדרך המקובלת ב-Access להגשת נתוני שרת-קובץ מאז המהדורה הראשונה. מהדורות קודמות של Access אמנם איפשרו גישה לנתוני שרת-לקוח, אולם עיבודי שרת-לקוח של ממש מעולם לא היו קלים כבמהדורה זאת. האמנם יכולים קבצי ‎*.adp לשרת מקורות

נתונים של שרת-לקוח באותה דרך שקבצי mdb.* שירתו מקורות נתונים שרת-קובץ? התשובה לשאלה זאת תלויה במידת העניין ובגבולות הדמיון שלך. הדוגמאות הבאות עשויות להצית בך את ניצוץ היצירתיות.

מיון ועיצוב באמצעות דוחות, ועוד

בעזרת מעצב השאילתות, קל למדי לסנן, לצרף ולצבור נתונים, אולם לא ניתן לעשות בו הרבה במונחים של מצגות. זכור שלא ניתן אפילו למיין את הרשומות בתוך תצוגה. לעומת זאת בדוחות, כושר העיבוד מוגבל למדי, אולם אפשרויות המיון והעיצוב של נתונים למדפסות מצוינות (ואפילו ל-Web, אם מביאים בחשבון תמונות – ראה פרק 6).

תרשים 12.12 מראה כיצד להשתמש בתצוגות של פרויקט Access עם דוחות Access כך שישלימו זה את זה. הדוח ממיין רשומות לפי FamID. כמו כן הוא מעצב באופן מותנה את הצבע לתצוגת שורה לפי הערך של FamID: ערכים נמוכים מ-6 מופיעים בשחור, וערכים השווים או גדולים מ-6 מופיעים באדום. נוסף לכך, התצוגה מראה את האבזרים המקובלים של דוח, לרבות כותרת, קו מפריד בין כותרות העמודות לבין הערכים בעמודות, וכן עיצוב לכותרות עמודות וכותרת הדוח, המבדיל בינן לבין העיצוב של גוף הדוח.

MyUnionQuery

MyUnionQuery

FamID	Fname	Lname	Relation
1	Rick	Dobson	Me
2	Virginia	Dobson	wife
3	Glen	Hill	son
4	Tony	Hill	son
5	Shelly	Hill	daughter-in-law
6	Dee	Simmons	sister
7	Foster	Simmons	nephew
8	Diane	Simmons	grand neice
9	Rick	Simmons	grand nephew

Page: 1

תרשים 12.12: דוח המבוסס על התצוגה MyUnionQuery המופיעה בתרשים 12.11

מלבד הקוד לשאילתת האיחוד עבור התצוגה, לא נדרש תכנות נוסף בדוח זה. לאחר שבנית את התצוגה, הצב את שם התצוגה בהגדרת המאפיין Record Source של הדוח. להוציא הצבעים השונים המבוססים על ערכי FamID, שאר תכונות הדוח מקבלות את הגדרות ברירת המחדל לדוחות טבלאיים. הפקודה החדשה **עיצוב מותנה** (Conditional Formatting) בתפריט **עיצוב** (Format) מפשטת באורח דרמטי את משימת ההחלה המותנית של צבעים על תכולתן של תיבות טקסט (ראה פרטים בפרק 5). אפשרויות אלה מחייבות שימוש בשגרות אירוע Format. על ידי שילוב התצוגה והדוח תוכל להשתמש בכל אחד מהם לביצוע המשימה שהוא מותאם לה ביותר. הואיל ודיווח הוא אחד מהיתרונות הבולטים של Access, יכולתו של קובץ adp.* לעבד מקורות נתונים שרת-לקוח יכול לצמצם באופן דרמטי את העלות של הפקה והפצת דוחות אינפורמטיביים וקלים לקריאה בכל רחבי הארגון.

הוספת היפר-קישורים

במסדי נתונים של SQL Server לא קיימים נתונים מסוג **היפר-קישור** (Hyperlink) (כזכור, Jet תומך בסוג נותנים זה כהרחבה של סוג הנתונים **תזכיר** (Memo) – ראה פרק 1). עם זאת, עדיין תוכל להוסיף ולעקוב אחר היפר-קישורים בתוך טופסי פרויקטים של Access. לשם כך עליך לבצע ארבע פעולות:

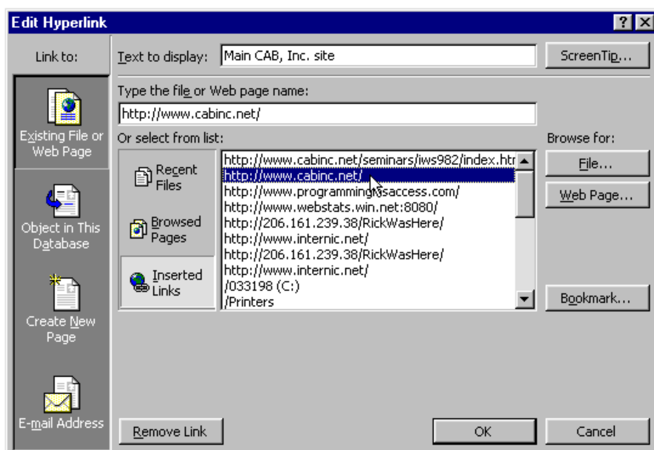
1. הקצה לעמודה בטבלה אחד מסוגי הנתונים הבאים: char, nchar, nvarchar או varchar. אלה הם שמות של סוגי נתונים קבועים ומחרוזות בעלות אורך משתנה, ב- Unicode או בתבנית שאינה Unicode.
2. פתח טופס בתצוגת עיצוב (Design) והפוך את הטבלה למקור הרשומות של הטופס.
3. הצב את השדה בעל סוג הנתונים מחרוזת באחד משדות הטופס.
4. הגדר למאפיין **הנו היפר-קישור** (IsHyperlink) של השדה ערך **כן** (Yes). זהו מאפיין חדש ב- Access 2000, המיועד במפורש להיפר-קישורים בטופסי פרויקטים של Access.

לאחר שיצרת את הטבלה והטופס, תוכל להוסיף לטבלה היפר-קישורים באמצעות הטופס, בעזרת הפקודה **הוספה** (Insert), **היפר-קישור** (Hyperlink). פקודה זאת פותחת תיבת דו-שיח להגדרת או עריכת ה-URL של ההיפר-קישור ואת הטקסט המוצג שלו. לאחר שההיפר-קישורים עוצבו, יכול המשתמש לעקוב אחריהם עד לאתרי ה-Web המתאימים על ידי לחיצה על הערך שבתוך שדה היפר-קישור. דפדפן ברירת המחדל יוביל אותו לאתר הנכון. לאחר מכן יוכל המשתמש לחזור ל- Access על ידי לחיצה על החץ Back (הקודם). פונקציונליות זאת התווספה ב- Access 97; כעת היא קיימת גם עבור מסדי נתונים SQL Server ו- MSDE באמצעות Access 2000.

תרשים 12.13 מציג את מסך MyLinks ב-Access, תוך כדי תהליך עריכת ההיפר-קישור. תוכל לדפדף אל אתר ה- Web הרצוי כך שלא תיאלץ להקליד את ה- URL. נוסף לכך,

תוכל להשתמש בתיבת הטקסט **טקסט שיוצג** (Text To Display) כדי להזין את הטקסט שיופיע במקום ה-URL. שים לב, שניתן להשתמש בתיבת דו-שיח זאת גם להגדרת היפר-קישורים אל קבצים השוכנים בשרת קבצים מקומי, ואפילו להתחבר לכתובות דואר אלקטרוני. האפשרות השנייה מפעילה את תוכנת ברירת המחדל לדואר אלקטרוני בתחנת עבודה של דפדפן, עם השם וכתובת הדואר האלקטרוני שציינת בקישור.





תרשים 12.13: השתמש בתיבת הדו-שיח **עריכת היפר-קישור** (Edit Hyperlink) כדי להזין ולערוך היפר-קישורים במסד נתונים משותף שרת-לקוח של היפר קישורים

עריכה והצגה של נתונים באמצעות טפסים

בפרויקטים של Access יש שתי הגדרות אפשריות למאפיין **סוג ערכת רשומות** (Recordset Type) לצורך עבודה עם טפסים ומקורות נתונים מקובלים, כמו למשל טבלאות של עובדים או לקוחות. הגדרות אלה ייחודיות לקבצי ‎*.adp‎. לקבצי ‎*.mdb‎ הרגילים הגדרה שונה במאפיין **סוג ערכת רשומות**.

כאשר אתה מעצב יישום שמשתמש בטפסים עם נתונים ממסד נתונים SQL Server או MSDE, יישום הלקוח תמיד עובד עם תמונה של הנתונים המקוריים בשרת. למרות שהנתונים המקומיים הם תמונה של הנתונים בשרת, ניתן לעדכן אותם בשרת. כדי לאפשר פונקציונליות מסוג זה, קיימות ב- Access 2000 ההגדרות **תמונה ניתנת לעדכון** (Updateable Snapshot) ו**תמונה** (Snapshot) עבור המאפיין **סוג ערכת רשומות** בפרויקטים של Access. תוכל לקבוע הגדרות אלה בכרטיסיה **נתונים** (Data) שבתיבת הדו-שיח **מאפיינים** (Properties) של הטופס. כמו כן תוכל לטפל במאפיין **סוג ערכת רשומות** בעזרת VBA או מאקרו של Access.

כאשר אתה מציב במאפיין **סוג ערכת רשומות** של טופס את ההגדרה **תמונה ניתנת לעדכון**, המשתמש יוכל לשנות את מקור הנתונים שביסודו של הטופס כאילו אותו קובץ נמצא בשרת קבצים מקומי. בעזרת המאפיין Lock של פקדים נפרדים, תוכל לאפשר עריכה בררנית של קבוצת משנה של פקדים בטופס. נוסף לכך, תוכל לשלוט בסוג השינויים המותרים לביצוע בתחנות עבודה של לקוחות. ישנן הגדרות כן/לא נפרדות לאפשרויות **אפשר עריכה** (Allow Edits), **אפשר מחיקות** (Allow Deletions) **ואפשר תוספות** (Allow Additions). כל שינוי שתערוך מתוך קובץ של פרויקט Access יופץ אל השרת. לכל משתמש בפרויקט Access צריך להיות עותק נפרד של קובץ ה-adp.*, למרות שכל משתמש פונה אל אותו מסד נתונים שרת-לקוח. למרות ששינויים בנתונים מופצים מתחנות העבודה של הלקוח אל מסד הנתונים של השרת, כדי לצפות בשינויים שערכו אחרים על המשתמש לבחור באפשרות **רענן** (Refresh) מתפריט **רשומות** (Records) כדי לראות שינויים שנערכו בידי משתמשים אחרים.

תרשים 12.14: השתמש בטפסים במסגרת פרויקטים של Access כדי להכניס שינויים בנתונים

תרשים 12.14 מציג דוגמת טופס בפרויקט Access. הטופס מבוסס על נתונים משרת מרוחק. מתחת לטופס מוצג קטע מגיליון המאפיינים של הטופס, החושף את שתי ההגדרות האפשריות למאפיין **סוג ערכת רשומות** (Recordset Type). הואיל וההגדרה שבתוקף היא **תמונה ניתנת לעדכון** (Updateable Snapshot), המשתמש יוכל להחליף את מקור הנתונים שביסודו של הטופס. למעשה, בגלל הגדרות ה**כן** לאפשרויות **אפשר עריכה**, **אפשר מחיקות** ו**אפשר תוספות**, הוא יוכל לבצע את השינויים משלושת הסוגים הללו במסד הנתונים. באפשרותך למנגנון שימוש באפשרויות אלה על ידי הצבת ההגדרה **תמונה** (Snapshot) במאפיין **סוג ערכת רשומות**.

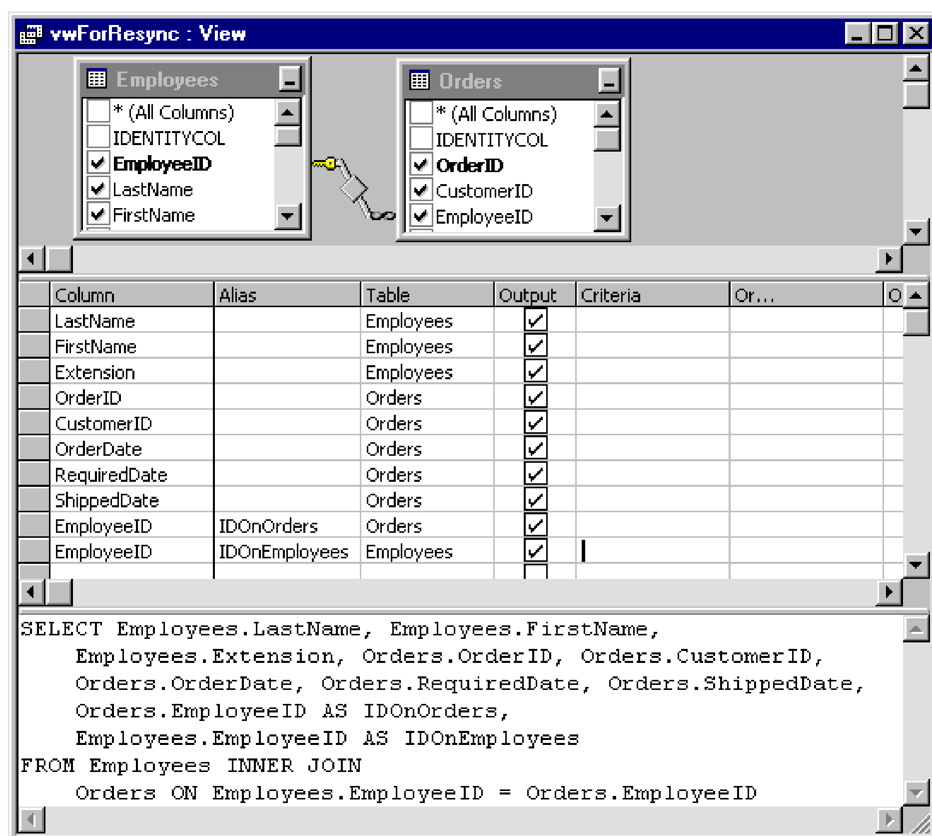
הטופס הוכן בעזרת אשף הטפסים האוטומטיים (בתוספת עריכה מינימלית). המעבר לרשומה הבאה גורם להעברת כל פעולת עריכה שביצע המשתמש בשדות בחזרה לשרת. אם הגדרת המאפיין **סוג ערכת רשומות** היא **תמונה**, Access מגיב בהודעה שקבוצת הרשומות אינה ניתנת לעדכון. הלחצן המסומן ב-X מאפשר למשתמש לבטל פעולת טעינה ארוכה של רשומות רבות אל תחנת עבודה מקומית.

סינכרון מחדש של טופס באמצעות נתוני יחיד לרבים

כאשר המקור של טופס בפרויקט Access מבוסס על ערכת רשומות שמתקיימים בה קשר גומלין של יחיד לרבים בין שתי טבלאות, ניתן לאכלס באופן אוטומטי את כל השדות בצד היחיד על ידי הגדרת המפתח הזר בצד הרבים. תכונה זאת שימושית במיוחד כאשר מוסיפים רשומות חדשות. לדוגמה, אם עתה עוסק בעיצוב טופס המייצג נתוני עובד ומכירות, היחס יחיד לרבים נגזר מהעובדה שכל עובד יכול לבצע פעולות מכירה רבות. עליך רק להזין את זיהוי העובד עבור טבלת המכירות, ולהשלים את הרשומה. פעולה זאת תגרום לאכלוס אוטומטי של כל שדות העובד בצד האחד של מקור הרשומות שברקע. לאחר מכן עליך להזין את נתוני המכירות בצד הרבים של מקור הרשומות. באותה קלות תוכל לעדכן את נתוני העובד גם ברשומה קיימת.

כדי להפוך תכונה זאת לזמינה, יש להגדיר את המאפיין **סוג ערכת רשומות** בתור **תמונה ניתנת לעדכון**. הצב את שם הטבלה בצד הרבים של קשר הגומלין בתור המאפיין **טבלה ייחודית** (Unique Table). כמו כן עליך להציב כהגדרת המאפיין **פקודת Resync** (Resync Command) מחרוזת המייצגת משפט SQL שמאפשר את הסינכרון מחדש. להלן נציג תחבולה ליצירת משפט זה בשיטה מקוצרת.

תרשים 12.15 מראה את תצוגת העיצוב של יחיד לרבים עבור התצוגה vwForResync. החלונית העליונה מראה את קשר הגומלין יחיד לרבים בצורת קו המחבר בין הטבלאות Employees ו-Orders. שים לב שתצוגה זאת מקשרת בין הטבלאות בעזרת המפתח הראשי בטבלה Employees ובעזרת המפתח הזר המקביל (EmployeeID) בטבלה Orders. החלונית התחתונה מציגה את קוד SQL של התצוגה. המאפיין **פקודת Resync** של הטופס מחייב עריכת שינוי פשוט במשפט זה.



תרשים 12.15: תצוגה זו מייצגת קשר גומלין של יחיד לרבים כמו זה המשמש לסינכרון מחדש של טופס, כאשר משפט SQL הוא הבסיס להגדרת המאפיין **פקודת Resync**

תרשים 12.16 שלהלן מציג טופס מייד לאחר הזנת הערך 9 לתיבת הטקסט EmployeeID. מדובר בטופס אוטומטי רגיל (AutoForm), עם שינויים מזעריים. לחיצה על בורר הרשומות (record selector) מעלה נתונים אל תוך כל שאר שדות העובדים בטופס. הזנת ערך חדש בתיבת הטקסט EmployeeID ולחיצה על בורר הרשומות תעלה באופן אוטומטי נתוני עובדים המתאימים לערך ID החדש. זהו יתרונה של פונקציית הסינכרון מחדש.

תרשים 12.17 מציג את הגדרות הטופס ההופכות את פעולת הסינכרון האוטומטית לזמינה. הגדרת המאפיין **סוג ערכת רשומות** (Recordset Type) של הטופס היא **תמונה ניתנת לעדכון** (Updateable Snapshot), והמאפיין **טבלה ייחודית** (Unique Table) שלו מצביע אל צד הרבים של ערכת הרשומות שברקע, כלומר, הטבלה Orders. ההגדרה **פקודת Rsync** (Resync Command) המוגדלת זהה בדיוק לקוד SQL של הטופס שבבסיס, בתוספת שורה אחת: *WHERE Orders.OrderID=?*. זהו השלב הסופי המשלים את תהליך הפיכתו של טופס לזמין בעזרת פונקציית הסינכרון מחדש.

vwForResync

EmployeeID	9
LastName	Dodsworth
FirstName	Anne
Extension	452
OrderID	11083
CustomerID	
OrderDate	
RequiredDate	
ShippedDate	

Record: 832 of 832

תרשים 12.16: טופס המדגים סינכרון מחדש מול מסד נתונים מרוחק

Form

Format | Data | Event | Other | All

Record Source	vwForResync
Filter	
Order By	
Allow Filters	Yes
Allow Edits	Yes
Allow Deletions	Yes
Allow Additions	Yes
Data Entry	No
Recordset Type	Updatable Snapshot
Max Records	10000
Server Filter	
Unique Table	Orders
Resync Command	SELECT Employees.La
Input Parameters	

Zoom

```
SELECT Employees.LastName, Employees.FirstName,
   Employees.Extension, Orders.OrderID,
Orders.CustomerID,
   Orders.OrderDate, Orders.RequiredDate,
Orders.ShippedDate,
   Orders.EmployeeID AS IDOnOrders,
   Employees.EmployeeID AS IDOnEmployees
FROM Employees INNER JOIN
   Orders ON Employees.EmployeeID =
Orders.EmployeeID
WHERE Orders.OrderID = ?
```

OK | Cancel | Font...

תרשים 12.17: הכרטיסיה **נתונים** (Data) בגיליון המאפיינים של הטופס; ההגדרות **תמונה ניתנת לעדכון ופקודת Resync** נחוצות לסינכרון מחדש

סוגיות של תכנות

אחת השיטות להרחבת היכולות של פרויקט Access היא להיעזר ב-SQL. שיטה זאת הולמת במיוחד כשמדובר בתצוגות ובשגרות מאוחסנות. בכמה מהדוגמאות הקודמות כבר ראית מה ניתן לעשות בעזרת SQL, אולם תוכל לנצל גם את בקיאותך ב- VBA וב-ADO. שאר הדוגמאות בפרק זה עוסקות ביישום הידע הזה בעבודה עם פרויקטים של Access, ואפילו בעבודה עם מסדי נתונים SQL Server ו- MSDE ללא ממשק המשתמש הנוח של פרויקטים של Access.

עבודה עם טפסים

למרות הפונקציונליות יוצאת הדופן של ממשק המשתמש בפרויקטים של Access מול מקורות נתונים מרוחקים, ניתן לבצע אוטומציה ופישוט של תהליכים על ידי תכנות של פיתוח פתרונות מותאמים אישית. אנו נבחן שלושה תחומים הנוגעים לפרויקטים של Access: פתיחת טופס, איתור רשומה והצגת שינויים שביצעו משתמשים אחרים.

פתיחת טופס

כאשר פותחים טופס בעזרת ממשק הפרויקטים של Access, הטופס מאכלס עותק מקומי של הנתונים המרוחקים בתחנת הלקוח. העותק המקומי הוא תמונה, נכונה לנקודת זמן מסוימת, של הטופס עם הנתונים המרוחקים. כאשר פותחים טופס באמצעות קוד, יש ליצור את העותק המקומי של הנתונים המרוחקים. אחד היתרונות בפתיחת טופס באמצעות קוד היא שניתן להקצות ערכים באופן דינמי למטמון המקומי המכיל את הנתונים המרוחקים. היישום שלך יוכל לעשות זאת מכיון שערכת הרשומות שהקצית לטופס בעזרת VBA גוברת על ההגדרה **מקור רשומה** (Record Source) שבגיליון המאפיינים של הטופס.

השיגרה הבאה בונה ערכת רשומות עבור טופס לפני פתיחתו. תחילה היא מגדירה הפניה למופע חדש של ערכת רשומות: היא מציבה את ההגדרה adUseClient במאפיין CursorLocation (מיקום הסמן) של ערכת הרשומות כדי לקבוע את מיקום הנתונים בטופס (כזכור, הטופס מקבל את הנתונים מהמטמון המקומי שבתחנת העבודה, ולא מהשרת המרוחק). כעת היא פותחת את ערכת הרשומות בעזרת משפט SQL המחלץ נתונים מהמקור המרוחק ומציב אותם בעותק המקומי. הדוגמה משתמשת בטופס המופיע בתרשים 12.14 לעיל. שורת הערה מראה משפט SQL המסוגל לדרוס את ברירת המחדל של מקור הרשומה של הטופס. לאחר יצירת העותק המקומי של הנתונים המרוחקים, השיגרה פותחת את הטופס ומציבה את העותק המקומי כהגדרת המאפיין Recordset (ערכת רשומות) של הטופס. מאפיין חדש זה מכיל את הפונקציונליות של המאפיין RecordsetClone; בנוסף, שינויים שנעשו בערכת הרשומות יופיעו באופן אוטומטי בטופס – המאפיין RecordsetClone יוצר עותק לקריאה-בלבד של ערכת הרשומות של הטופס. בדומה למאפיין RecordsetClone, המאפיין Recordset של הטופס זמין רק באמצעות הקוד.

```
Sub openForm()
Dim rst1 As ADODB.Recordset
' Establish a local recordset and populate it with values;
' can override property sheet settings.
    Set rst1 = New ADODB.Recordset
    rst1.CursorLocation = adUseClient
    rst1.Open "SELECT * FROM employees", _
        CurrentProject.Connection, adKeySet, _
        adLockPessimistic
' Optionally run with WHERE clause to demo override effect.
'    "SELECT * FROM employees WHERE employeeid>3"
' Open the form.
    DoCmd.openForm "frmemployees2"
' Assign recordset to the open form; can override a setting on the property sheet.
    Set Forms("frmemployees2").Recordset = rst1
End Sub
```



הערה:
ערכת הרשומות שנוצרת עבור מופע של ערכת רשומות תישאר זמינה כל עוד הטופס פתוח. סגירה ופתיחה מחדש ידנית של הטופס תגרום לטופס לחזור להגדרת **מקור רשומה** (Record Source) שבגיליון המאפיינים שלו.

איתור רשומה

בעבודה עם נתונים בטופס, אחת המשימות השכיחות היא איתור רשומה מסוימת. שתי השגרות הבאות מבצעות משימה זאת. השיגרה locateEmployee מנחה את המשתמש להזין ערך של זיהוי עובד, ומעבירה ערך זה אל השיגרה השנייה, findByID. השיגרה השנייה מפעילה את השיטה FindRecord של האובייקט DoCmd כדי לאתר את הרשומה ולמקם את הטופס ברשומה החדשה.

```
Sub locateEmployee()
' Ask for employee ID and pass it on to findByID.
    employeeNumber = InputBox("Type the ID for the employee you want", _
        "Programming Microsoft Access 2000")
    findById CLng(employeeNumber)
End Sub

Sub findById(eid As Long)
On Error GoTo findByIdTrap
' Set focus to employee ID field and launch find.
    Forms("frmemployees2").EmployeeID.SetFocus
    DoCmd.FindRecord eid
```

```
findByIdExit:
' Report mismatch before exiting.
    If Forms("frmemployees2").EmployeeID <> eid Then
        MsgBox "No employee with ID " & eid & ".", _
            vbExclamation, "Programming Microsoft Access 2000"
    End If
    Exit Sub

findByIdTrap:
    If Err.Number = 2450 Then
' Open form if it is closed and start find again.
        openForm
        Resume
    Else
        Debug.Print Err.Number, Err.Description
    End If
End Sub
```

השיגרה השנייה מזהה שתי בעיות אפשריות. ראשית, אם לא נמצאת רשומה תואמת, השיגרה מציגה הודעה ברוח זאת. שנית, אף אחת משתי השגרות אינה שגרת אירוע, לכן ניתן להפעילן גם מחוץ לטופס. אם הטופס סגור, השיגרה מזהה את הבעיה ופותחת את הטופס כדי לנסות פעם נוספת לאתר את רשומת היעד.

הצגת שינויים שביצעו משתמשים אחרים

Access הוא ביסודו סביבת פיתוח מרובת-משתמשים, ופרויקטים של Access מופעלים תכופות כדי לשרת מטרות מרובות-משתמשים. לכן דרושה לך שיטה לרענון המטמון המקומי, כדי שתוכל להציג עדכונים, תוספות ומחיקות שבוצעו בידי אחרים. השיגרה הבאה מבצעת משימה זאת על ידי איכלוס מחדש של ערכת הרשומות המקומית של הטופס בנתוני מקור הנתונים המרוחק. בדומה לדוגמאות הקודמות, דוגמה זאת מבוססת על הטופס המופיע בתרשים 12.14.

```
Sub requeryRemoteRestoreID()
Dim int1 As Integer
' Turn off screen updates and save employee ID.
    DoCmd.Echo False
    int1 = Forms("frmemployees2").EmployeeID
' Requery local recordset from the server.
    openForm
' Reposition to employee ID before requery and
' restore screen updating.
    Forms("frmemployees2").EmployeeID.SetFocus
    DoCmd.FindRecord int1
    DoCmd.Echo True
End Sub
```

השיגרה מאכלסת מחדש את מטמון הנתונים המקומי על ידי הפעלת השיגרה openForm (שעסקנו בה קודם). אם הטופס פתוח, הקריאה רק מאכלסת מחדש את המטמון. הואיל ואיכלוס מחדש של ערכת הרשומות של טופס ממקם באופן אוטומטי את הטופס ברשומה הראשונה, השיגרה שומרת את מיקום הרשומה הנוכחית לפני שהיא מאכלסת מחדש את המטמון המקומי בנתוני מחסן הנתונים המרוחק. לאחר איכלוס המטמון מחדש השיגרה מחפשת בין הרשומות כדי לאתר את הרשומה הנוכחית הקודמת. פעולה זאת משחזרת את מיקום הרשומה הישנה, אם היא עדיין זמינה (כלומר, אם לא נמחקה). אם משתמש אחר מחק את הרשומה שהיתה הרשומה הנוכחית קודם, הטופס יציג את הרשומה הראשונה במטמון המקומי.

עבודה עם מודולים עצמאיים

ההדגשה החוזרת ונשנית על ADO לאורך ספר זה תשרת אותך היטב כאשר תעסוק בפיתוח פתרונות מותאמים אישית מול מקורות נתונים MSDE ו- SQL Server. פרק 4 מכיל מספר דוגמאות לטיפול במקורות נתונים מרוחקים. בסעיף זה נבחן מחדש סוגיות תכנות מסוימות בהקשר עם פרויקטים של Access ו- MSDE.

פתיחת טבלה

הדוגמה הבאה משלבת בין פתיחת טבלה במקור נתונים מרוחק לבין כתיבת התוצאות בחלון Immediate (מיידי), המשמש לצורך העניין כמדריך טלפונים. לשם כך הדוגמה משתמשת באובייקטים Connection (חיבור) ו- Recordset (ערכת רשומות) של ADO. השיגרה פותחת ביצירת מופע חדש של האובייקט Connection של ADO. לאחר מכן היא מגדירה מחרוזת חיבור ומשתמשת בה כדי ליצור חיבור עם מסד הנתונים NorthwindCS בשרת cab2200. בשלב הבא היא פותחת ערכת רשומות קדימה-בלבד לקריאה-בלבד, המבוססת על הטבלה Employees במסד הנתונים NorthwindCS. ערכת רשומות מסוג זה קבילה בדוח המדפדף פעם אחת ויחידה ברשומות של ערכת רשומות. השיגרה מדפיסה את מדריך הטלפונים של עובדים בחלון Immediate בעזרת לולאת Do כדי לעבור על פני הרשומות העוקבות, לפי הסדר.

```
Sub openTableOnRemoteServer()
Dim cnn1 As ADODB.Connection
Dim rst1 As ADODB.Recordset
' Open connection to NorthwindCS database on cab2200 server.
    Set cnn1 = New ADODB.Connection
    strCnn = "Provider=sqloledb;" & _
        "Data Source=cab2200;" & _
        "Initial Catalog=NorthwindCS;" & _
        "User Id=sa;Password=;"
    cnn1.Open strCnn
```

```
' Open employee table with a forward-only, read-only recordset;
' this type is OK for a single pass through the data.
    Set rst1 = New ADODB.Recordset
    rst1.CursorType = adOpenForwardOnly
    rst1.LockType = adLockReadOnly
    rst1.Open "employees", cnn1, , , adCmdTable

' Print an employee telephone directory.
    Do Until rst1.EOF
        Debug.Print rst1.Fields("FirstName") & " " & _
            rst1.Fields("LastName") & " has extension " & _
            rst1.Fields("Extension") & "."
        rst1.MoveNext
    Loop

' Close the connection and recover the resource.
    cnn1.Close
    Set cnn1 = Nothing
End Sub
```

סביבת הבדיקות של פרק זה כוללת שרת MSDE על מחשב מקומי בשם CABXLI, נוסף למנהל מסד הנתונים SQL Server השוכן במחשב ששמו cab2200. התחביר של הפניה לשרתים מקומיים שונה במקצת מהתחביר להפניה אל שרתים מרוחקים. במקום לציין שרת מוגדר בשמו, תוכל לציין פשוט (local). מכיון שגם במחשב המקומי שלי מותקן מסד הנתונים NorthwindCS, אין צורך בשינויים נוספים בשיגרה. שיגרה זאת זהה לשיגרה הקודמת, למעט הקטע האחד המופיע להלן. שים לב לשם השרת החדש. בפועל, אולי רצוי שהמשתמשים יקבעו לעצמם שם חשבון כניסה, או שישתמשו בחשבון מוגבל בעל פחות זכויות מאשר sa.

```
' Open connection to NorthwindCS database on the local server.
    Set cnn1 = New ADODB.Connection
    strCnn = "Provider=sqloledb;" & _
        "Data Source=(local);" & _
        "Initial Catalog=NorthwindCS;" & _
        "User Id=sa;Password=;"
    cnn1.Open strCnn
```

תוכל לעבד תצוגות באמצעות אותו תחביר המשמש אותך לטבלאות. עליך רק לתחום את שם התצוגה במרכאות כפולות, כפי שנהוג בטבלאות. עדיין תידרש להשתמש בהגדרה adCmdTable לפרמטר Option. התקליטור המצורף מכיל דוגמה בשם openViewOnRemoteServer, המציגה גישה זאת.

פתיחת שיגרה מאוחסנת

שגרות מאוחסנות יכולות להחזיר ערכות רשומות. החזרת ערכות רשומות בעזרת שגרות מאוחסנות במקום בעזרת טבלאות תעניק תוספת גמישות ליישומיך מכיון שהשימוש בשגרות מאוחסנות ישחרר אותך מהתלות בטבלאות במקור הנתונים הנוכחי, או אפילו עותקים מדויקים של הטבלאות במקור הנתונים הנוכחי. בעזרת שגרות מאוחסנות תוכל לסנן רשומות, לחשב ערכים חדשים ולצרף ערכי שדה לאורך הרשומות בטבלה.

השיגרה הבאה מפעילה את השיגרה המאוחסנת ששמה Ten Most Expensive Products ומדפיסה בחלון Immediate את שמות עשרת המוצרים היקרים ביותר ומחיריהם. מכיון ששם השיגרה מכיל תווי רווח, חובה לתחום אותו בסוגריים מרובעים. שים לב גם לכך שהשיגרה משתמשת ב- AdCmdStoredOProc כארגומנט Options. דבר זה מורה למעבד ADO לצפות לשם-שיגרה המכיל SQL, אך לא למשפט SQL של ממש. הואיל והשיגרה תמיד מעבדת 10 רשומות, השיגרה להדפסת הרשומות משתמשת בלולאת For החוזרת על עצמה מ-1 עד 10. בכל שאר המובנים, השיגרה להדפסת הקבוצה המוחזרת של שיגרה מאוחסנת זהה לשיגרה להדפסת טבלה שלמה.

```
Sub openProcedureOnRemoteServer()
Dim cnn1 As ADODB.Connection
Dim rst1 As ADODB.Recordset
Dim int1 As Integer
' Open connection to NorthwindCS database on cab2200 server.
   Set cnn1 = New ADODB.Connection
   strCnn = "Provider=sqloledb;" & _
      "Data Source=cab2200;" & _
      "Initial Catalog=NorthwindCS;" & _
      "User Id=sa;Password=;"
   cnn1.Open strCnn
' Open employee table.
   Set rst1 = New ADODB.Recordset
   rst1.CursorType = adOpenForwardOnly
   rst1.LockType = adLockReadOnly
   rst1.Open "[Ten Most Expensive Products]", cnn1, , , adCmdStoredProc
' Print prices for 10 products.
   For int1 = 1 To 10
      Debug.Print rst1.Fields(0) & " has a unit price of $" & rst1.Fields(1) & "."
      rst1.MoveNext
   Next int1
' Close the connection and recover the resource.
   cnn1.Close
   Set cnn1 = Nothing
End Sub
```

הצבת ערכים במאפיין CursorLocation

ההגדרות שתציב ב-CursorLocation עשויות להשפיע באורח מהותי על רמת הביצועים, אפילו כאשר מדובר בנפח בינוני של מקורות טבלאיים. הדוגמה הבאה מראה את הדבר בעזרת מקור רשומה המכיל מעט מעל 19,000 רשומות. השיגרה יוצרת את ערכת הרשומות בעזרת תצוגה המבוססת על המכפלה הקרטזית של הטבלאות Employees ו-Order Details. שם התצוגה הוא vwLargeView.

הדוגמה פותחת את מקור ערכת הרשומות בעזרת adUseClient או adUseServer כהגדרת המאפיין CursorLocation שלה. ההגדרה adUseServer גורמת לשיגרה להתקדם דרך הרשומות שבשרת, בזו אחר זו. ההגדרה adUseClient מעבירה את הרשומות אל תחנת העבודה המקומית, כך שהשגרות יוכלו לגשת אל הרשומות דרך תחנת עבודה מקומית מבלי לחזור אל השרת עבור כל רשומה.

השיגרה הראשונה בדוגמה מנחה את המשתמש להזין את הגדרת CursorLocation הרצויה. השיגרה השנייה מכינה דוח ומדפיסה אותו בחלון **Immediate**. היא מדפיסה גם את מיקום וסוג הסמן, וכן את שעת ההתחלה, שעת הסיום ומשך הזמן של המשימה. תוכל להתאים את התבנית הכללית הזאת לבחינת צירופים שונים בין הגדרות של מסדי נתונים לבין מקורות הנתונים שלך.

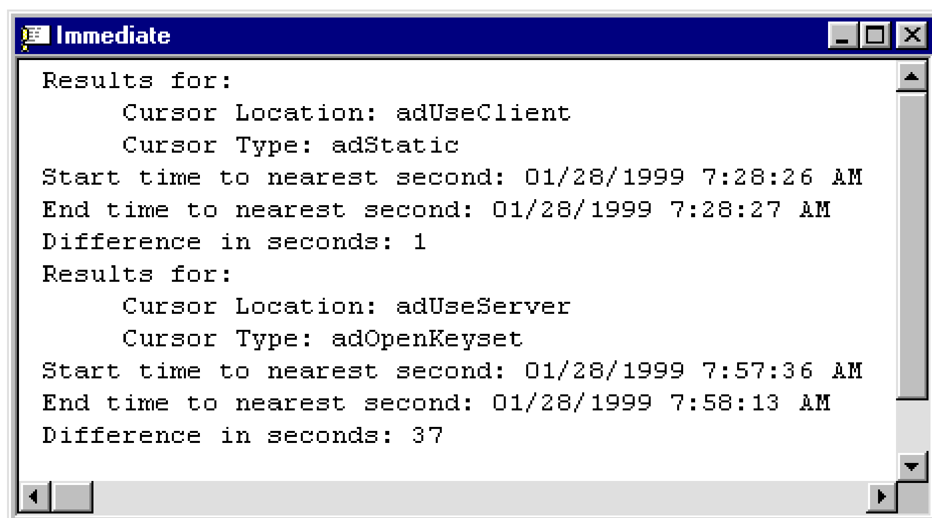
```
Sub openRemoteWithCursorLocation()
Dim cnn1 As ADODB.Connection
Dim rst1 As ADODB.Recordset
Dim start As Date, done As Date
' Open connection to NorthwindCS database on cab2200 server.
    strCnn = "Provider=sqloledb;" & _
        "Data Source=cab2200;" & _
        "Initial Catalog=NorthwindCS;" & _
        "User Id=sa;Password=;"
    Set cnn1 = New ADODB.Connection
    If MsgBox("Use local cursor?", vbYesNo, _
        "Programming Microsoft Access 2000") = vbYes Then
        cnn1.CursorLocation = adUseClient
    Else
        cnn1.CursorLocation = adUseServer
    End If
    cnn1.Open strCnn

' Open vwLargeView view; create the view in the remote
' database server before running the procedure.
' The sample uses the Cartesian product of the
' Employees and Order Details tables.
' Notice that the connection setting for CursorType silently
' overrides the recordset property setting.
```

```
    Set rst1 = New ADODB.Recordset
    rst1.CursorType = adOpenKeyset
    rst1.LockType = adLockOptimistic
    rst1.Open "vwLargeView", cnn1, , , adCmdTable
    start = Now
    Do Until rst1.EOF
        temp = rst1.Fields(1)
        rst1.MoveNext
    Loop
    done = Now
    reportResults cnn1.CursorLocation, rst1.CursorType, start, done
End Sub

Sub reportResults(cloc As Integer, ctype As Integer, _
    startedAt As Date, endedAt As Date)
    Debug.Print "Results for: "
    Select Case cloc
        Case adUseServer
            Debug.Print "     Cursor Location: adUseServer"
        Case adUseClient
            Debug.Print "     Cursor Location: adUseClient"
        Case Else
            Debug.Print "     Faulty Cursor Location setting"
    End Select
    Select Case ctype
        Case adOpenForwardOnly
            Debug.Print "     Cursor Type: adOpenForwardOnly"
        Case adOpenKeyset
            Debug.Print "     Cursor Type: adOpenKeyset"
        Case adOpenDynamic
            Debug.Print "     Cursor Type: adOpenDynamic"
        Case adOpenStatic
            Debug.Print "     Cursor Type: adStatic"
    End Select
    Debug.Print "Start time to nearest second: " & startedAt
    Debug.Print "End time to nearest second: " & endedAt
    Debug.Print "Difference in seconds: " & DateDiff("s", & startedAt, endedAt)
End Sub
```

תרשים 12.18 מציג את התוצאות של הפעלת השגרות, תחילה עם ההגדרה adUseClient ולאחר מכן עם ההגדרה asUseServer. מעבר הלולאה בין הרשומות נמשך שנייה אחת בלבד עם סמן מקומי, אולם נדרשו 37 שניות למעבר הלולאה דרך הרשומות בעזרת סמן בצד-השרת. התוצאות עשויות להשתנות, בהתאם למגוון רחב של גורמים, לכן רצוי לבדוק את העניין עם הגדרות נוספות מול מסדי הנתונים המשמשים אותך.



תרשים 12.18: דוגמת פלט המראה את תוצאות העיבוד של אותו מקור נתונים, פעם עם ההגדרה adUseClient ופעם עם ההגדרה adUseServer

דוגמה זאת חשובה מכמה טעמים. ראשית, רמת הביצוע משתנה מאוד בהתאם להגדרה. שנית, ניתן לראות באמצעותה ש-ADO משנה את ההגדרות שלך אם קיימת ביניהן התנגשות. לדוגמה, התוכנית מציבה adOpenKeyset כהגדרה של סוג הסמן, אולם הגדרה זאת מתנגשת עם ההגדרה adUseClient, לכן מתרגם ADO משנה בשקט (ללא התראה) את הגדרת סוג הסמן ל- adStatic. דבר דומה מתרחש גם במקרים נוספים. שלישית, דוגמה זאת מציגה תבנית פשוטה שתוכל להתאימה בנקל לבדיקת הגדרות של מסדי נתונים ביישומיך המותאמים אישית.

13

Access 2000 וה-Web

Microsoft Access 2000 ממשיך במסורת הכפולה של טכנולוגיית אינטרנט חדשנית ושיטות שימושיות ופשוטות לפיתוח פתרונות בתחום מסדי הנתונים ל-Web. Access 2000 מכיל דפי גישה לעמודים, פקדי ActiveX חדשים, קישוריות מסד נתונים משופרת, ו-HTML דינמי (DHTML). יכולות חדשות אלו מציינות את קיצו של אשף פרסום התכנים ב-Web מתוך Access 97, שהכיל ממשק לפקד של מתאר HTML. פקד זה לא היה אלא טכנולוגיית ביניים, עד שיושלם פיתוחה של שפת DHTML מבוססת על סטנדרטים.

פרק זה סוקר את הטכנולוגיות המסורתיות של Access ל-Web, ולאחר מכן יעסוק בטכנולוגיית ההיפר-קישורים, לרבות האובייקט Hyperlink וסוג הנתונים **היפר-קישור** (Hyperlink). הפרק מוקדש ברובו לדפי גישה לנתונים, המפשטים את תהליכי הדיווח וקשרי הגומלין עם מסדי נתונים דרך ה- Web. סביבת הפיתוח לדפי גישה לנתונים עשויה לשמש גם בתפקיד מארח לרכיבי Web ב- Microsoft Office 2000.

גישות מסורתיות

טכנולוגיות Access המסורתיות ל-Web מתאימות במיוחד לפיתוח פתרונות בנפח פעילות נמוך (כלומר, עד 10,000 מבקרים ביום) ל-Web ולמסדי נתונים. מכיון שבמקרים מסוימים יש אי-התאמה בין גרסאות דפדפנים שונות של יצרנים שונים, דווקא הגישות המסורתיות הן הבטוחות, וגם קל למדי להשתמש בהן. Access 2000 כולל שלוש גישות מסוג זה:

- פרסום גליונות נתונים בתבנית HTML, IDC/HTX ו- ASP.
- שימוש בטפסי HTML. אין ב- Access סביבת פיתוח מוכללת לטפסי HTML, אולם תוכל להתאים את טפסי HTML הישנים שלך לטכנולוגיית OLE DB/ADO (ActiveX Data Objects).
- פרסום דוחות בתבנית תמונה (snapshot) בתוך תיקיית FTP. שיטה זאת מפשטת את הגישה לדוחות Access בעזרת דפדפני Netscape.

פרסום גליונות נתונים

כדי לייצא גיליון נתונים של Access, כמו למשל טבלה, בחר בפקודה **יצא** (Export) מתוך תפריט **קובץ** (File) בחלון מסד הנתונים. פקודה זאת (המפרסמת גם גליונות נתונים שבבסיס טפסים ודוחות) מכילה חלק מהפונקציונליות שהיתה בעבר לאשף פרסום התכנים ב-Web המיושן. הפקודה מפרסמת גליונות נתונים בתבנית HTML, IDC/HTX ו- ASP. תבנית HTML אמנם סטטית, אולם קל לערוך אותה בעזרת עורכי HTML הרגילים. שתי התבניות האחרות דינמיות. הם משתמשות בשרת כדי לכתוב תכני HTML כך שישקפו את התוצאות העדכניות ביותר בגיליון הנתונים.

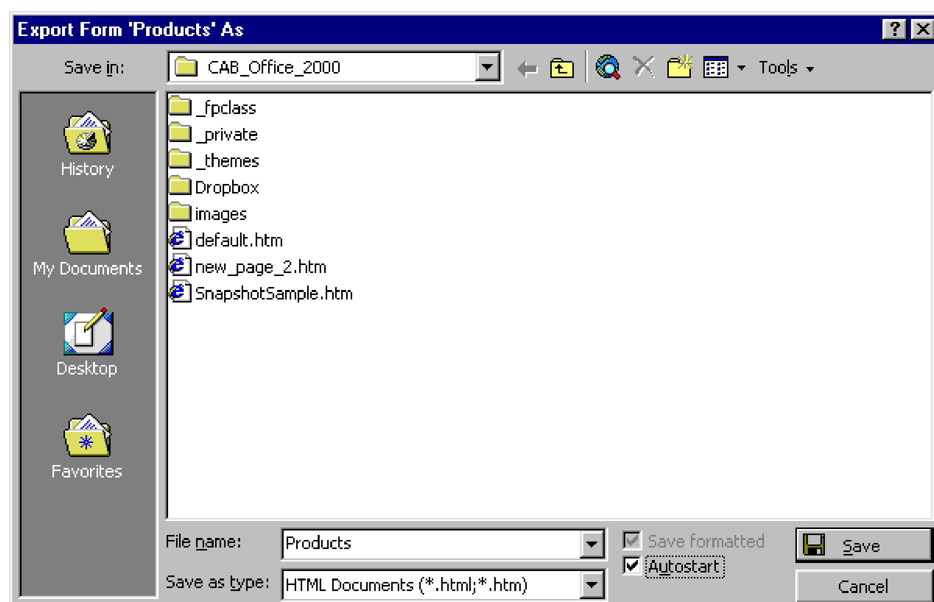
תבנית IDC/HTX מיועדת למעשה לשתי המהדורות הראשונות של Microsoft Internet Information Server (IIS), אולם היא מסוגלת לפעול גם עם מהדורות חדשות יותר. תבנית ASP פועלת רק עם המהדורות החדשות יותר, אולם מאפשרת שילוב של Microsoft Visual Basic Scripting Edition ו- Microsoft Jscript בקוד HTML. לא כדאי לבחור באופן אוטומטי באחת התבניות הדינמיות. אם אתה עובד עם גליונות נתונים גדולים או בסביבה מרובת-משתמשים, ואם התכנים בגליונות הנתונים אינם משתנים תדיר, או אם הזמן אינו גורם חשוב, ייתכן בהחלט שתבנית HTML הסטטית תתאים לך יותר, הואיל ופרסום דפים סטטיים מהיר יותר.

פרסום בתבנית HTML

כדי לפרסם גיליון נתונים בתבנית HTML, בחר טבלה, שאילתה, טופס או דוח בחלון מסד הנתונים, ולאחר מכן בחר באפשרות **יצא** מתפריט **קובץ**. בתיבת הדו-שיח **יצא**, השתמש ברשימה הנפתחת **שמור ב** (Save In) לבחירת מיקום לקובץ. אם אתה מפרסם ברשת ארגונית (אינטרא-נט), המיקום יכול להיות תיקיה באתר. אם אתה מפרסם

בשרת Web מרוחק, תוכל להשתמש בכל תיקיה מקומית לאיסוף דפי ה-HTML, ולאחר מכן להעביר את דפים לשרת המרוחק בעזרת הפקודה **File**, **Import** ב- FrontPage 2000. תוכל להשתמש גם בנתיבים אחרים, כמו למשל פרוטוקול FTP.

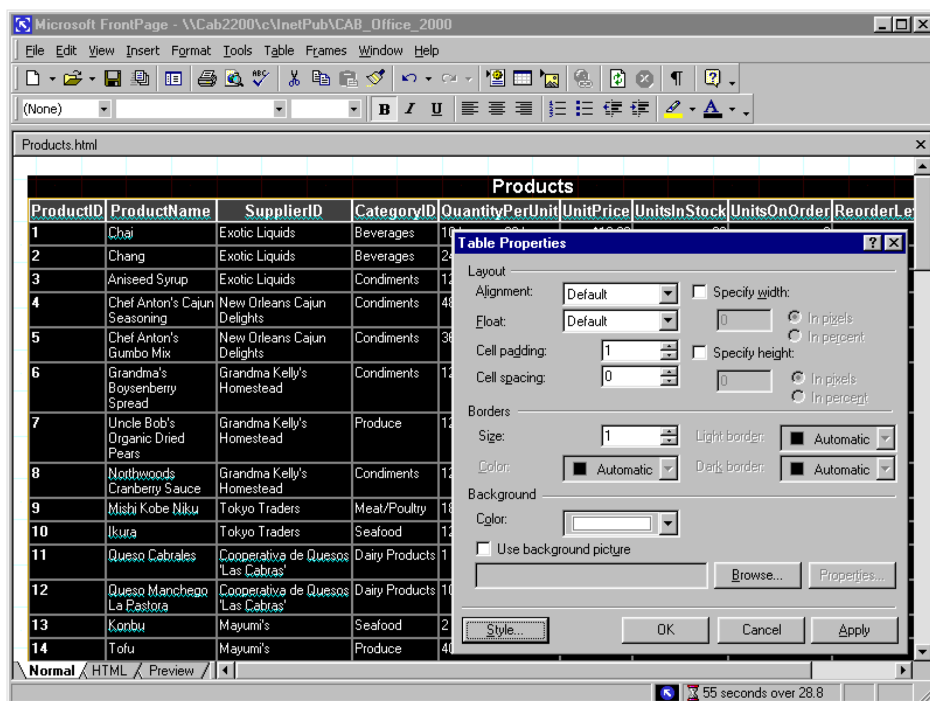
תרשים 13.1 מציג את תיבת הדו-שיח **יצא** לפרסום גיליון הנתונים שבבסיס הטופס Products מתוך מסד הנתונים Northwind. תיבת הרשימה הנפתחת **שמור כסוג** (Save As Type) מראה את תבנית HTML. התיבה **שמור ב** מגדירה את התיקיה CAB_Office_2000 כתיקיית הבסיס של אתר ברשת ארגונית מקומית בעל שם זהה. תיבת הסימון **אתחול אוטומטי** (Autostart) נבחרה, לכן הדף יופיע ברגע שיפורסם. דבר זה יכול לגרום להפעלת דפדפן להיפתח כאשר גיליון הנתונים Products פתוח.



תרשים 13.1: תיבת הדו-שיח **יצא**

כאשר לוחצים על **שמור** (Save), מופיעה תיבת הדו-שיח **אפשרויות פלט HTML** (HTML Output Options). כאן תוכל לציין תבנית לדף המכיל את גיליון הנתונים. אם הגדרת ערכת נושא לדף default.htm, תוכל להפנות אל דף זה. אם לא, תוכל להפנות אל קובץ מקומי עם העיצוב שישמש אותך באתר המרוחק.

אחד היתרונות של תבנית HTML הוא שהפעולה שעליך לבצע היא עריכת טבלת HTML. תרשים 13.2 מציג את גיליון הנתונים Products שפורסם ב- FrontPage 2000, ואת תיבת הדו-שיח **Table Properties**, המציגה חלק מאפשרויות העיצוב, לרבות הגנת תאים, מרווח בין תאים וצבע או תמונת רקע. כמו כן תוכל להגדיר מאפיינים לתאים, שורות ועמודות כך שיציינו אפשרויות עיצוב אחרות. באפשרותך לבטל עיצובים מותאמים אישית ב-FrontPage, ואפילו לבחור עמודות או שורות נפרדות, ולמחוק אותן באופן בררני.



תרשים 13.2: לאחר טעינת גליונות הנתונים אל תוך FrontPage, תוכל לערוך את טבלת HTML שתיווצר

פרסום בתבנית IDC/HTX

כדי לפרסם גיליון נתונים באחת משתי התבניות הדינמיות, דרוש לך שם מקור נתונים (DSN) בקישוריות מסד נתונים פתוחה (ODBC). על שם מקור הנתונים לשכון בשרת ה-Web, לצד מסד הנתונים שאליו הוא מפנה. גישת DSN של קבצים היא לכאורה גמישה יותר, הואיל וניתן לשלוח את שם מקור הנתונים לשרת, אולם במקרים רבים, ספקי שירותי אינטרנט מחילים כללים מיוחדים שהופכים את הפעלת הקבצים הללו למסובכת יותר. אצל ספקי השירותים נהוג להפיק דרך קבץ DSN אחד או יותר לכל חשבון של אתר Web עם זכויות שימוש במסדי נתונים. לכן פעמים רבות קל יותר להשתמש בשמות מקור נתונים של המערכת ולא באלה של קבצים. בכל מקרה, שמות מקור נתונים של המערכת עדיפים מבחינת רמת הביצועים.

לאחר שבחרת מקור גיליון נתונים בחלון מסד הנתונים, ולאחר שבחרת **קובץ**, **יצא**, בחר באפשרות (*.ht.;*.idc) Microsoft IIS 1-2 בתיבת הרשימה **שמור כסוג**. בחר באפשרויות המתאימות לך, באותו אופן שנהוג עבור קובץ HTML. לחיצה על **שמור** תגרום לפתיחת תיבת הדו-שיח **אפשרויות פלט HTX/IDC** (HTX/IDC Output Options) (המוצגת בתרשים 13.3). השתמש בלחצן **עיון** (Browse) לבחירת התבנית שתעצב את הדף שיחזיר את התוצאות מקובץ IDC. כדי לפעול, קובץ IDC זקוק לשם DSN בשרת.

בתיבת הטקסט **שם מקור הנתונים** (Data Source Name), הזן את שם מקור נתונים שמנהל האתר שלך הקצה למשימה זאת. תיבת הדו-שיח כוללת גם שדות זיהוי משתמש וסיסמה, למקרה שמסד הנתונים בשרת ה- Web פועל עם אבטחה ברמת המשתמש (מקרים כאלה נדירים, במיוחד ביישומי אינטרנט).

HTX/IDC Output Options
HTML Template: c:\InetPub\CAB_Office_2000\default.htm
Browse...
Data Source Information
Data Source Name: Nwind2000
User to Connect As:
Password for User:
OK
Cancel

תרשים 13.3: תיבת הדו-שיח **אפשרויות פלט HTX/IDC**

קבצי IDC ו-HTX משלימים זה את זה. קובץ IDC מפעיל שאילתה מול מסד נתונים של Access בשרת ה- Web. קובץ HTX משתמש בהרחבות HTML כדי לעצב את הקבוצה המוחזרת בתבנית HTML, לצורך הצגה באמצעות דפדפן. אמנם ניתן לכתוב את הקוד של קובץ HTX בהרחבת HTML, אולם הקובץ מחזיר אל הדפדפן HTML טהור, כדי שכל דפדפן יוכל לקרוא את התוצאות. מכיון שקובץ IDC יכול לחשוף את הקבוצה המוחזרת שלו רק באמצעות קובץ HTML, תבנית זאת מתאימה לעיבוד גליונות נתונים. עם זאת, אין היא מתאימה לעיבוד טפסים, מכיון שתבנית ההחזרה של HTML אינה קלט מתאים עבור מרבית טפסי HTML.

כאשר מפעילים זוג של IDC/HTX מתוך שרת Web, הדפדפנים פונים אל קובץ IDC, שחייב לשכון בתיקיית שרת Web המסוגלת להפעיל scripts. קובץ IDC מפעיל באופן אוטומטי את קובץ HTX, אשר בתורו מעבד את הקבוצה המוחזרת של קובץ IDC ומשגר את התוצאות, בתבנית HTML, בחזרה לדפדפן.

פרסום בתבנית ASP

בדומה לקבצי IDC ו-HTX, קבצי ASP זקוקים להפניה אל DSN. בתיבת הדו-שיח **יצא**, בחר באפשרות **Microsoft Active Server Pages (*.asp)** מתיבת הרשימה **שמור כסוג**. בתיבת הדו-שיח **אפשרויות פלט עמודים של שרתי Microsoft Active** (Microsoft Active Server Pages Output Options) הזן שם של DSN. שאר השדות בתיבת הדו-שיח אופציונליים, ותלויים בהעדפות העיצוב שלך ובהגדרות האבטחה של מסד הנתונים (שדות אלה אופציונליים גם בתיבות הדו-שיח **IDC/HTX** ו-**HTML**).

תבנית ASP שונה מתבנית IDC/HTX בכך שיש רק קובץ אחד. קובץ ASP מבצע את השאילתה ומעצב את התוצאה לצפיה באמצעות הדפדפן. הקובץ מפעיל את השאילתה שלו ויוצר את ה-HTML בשרת, ולאחר מכן שרת ה-Web מעביר את דפי ה-HTML בחזרה אל הדפדפן. קוד ASP שלהלן כותב את גיליון הנתונים שבבסיס הטופס Products שבמסד הנתונים Northwind אל תוך דפדפן:

```
<HTML><HEAD>
<META HTTP-EQUIV="Content-Type"
   CONTENT="text/html;charset=windows-1252">
<TITLE>Products</TITLE>
</HEAD><BODY>
<%
If IsObject(Session("Nwind2000_conn")) Then
   Set conn = Session("Nwind2000_conn")
Else
   Set conn = Server.CreateObject("ADODB.Connection")
   conn.open "Nwind2000","",""
   Set Session("Nwind2000_conn") = conn
End If
%>
<%
If IsObject(Session("Products_rs")) Then
   Set rs = Session("Products_rs")
Else
   sql = "SELECT * FROM [Products]"
   Set rs = Server.CreateObject("ADODB.Recordset")
   rs.Open sql, conn, 3, 3
   If rs.eof Then
      rs.AddNew
   End If
   Set Session("Products_rs") = rs
End If
%>
<TABLE BORDER=1 BGCOLOR=#ffffff CELLSPACING=0 RULES=none>
<FONT FACE="Arial" COLOR=#000000><CAPTION><B>Products</B>
</CAPTION></FONT>
<THEAD><TR>
<TH BGCOLOR=#c0c0c0 BORDERCOLOR=#000000 >
<FONT SIZE=2 FACE="Arial" COLOR=#000000>ProductID</FONT></TH>
<TH BGCOLOR=#c0c0c0 BORDERCOLOR=#000000 >
<FONT SIZE=2 FACE="Arial" COLOR=#000000>ProductName</FONT></TH>
<TH BGCOLOR=#c0c0c0 BORDERCOLOR=#000000 >
<FONT SIZE=2 FACE="Arial" COLOR=#000000>SupplierID</FONT></TH>
<TH BGCOLOR=#c0c0c0 BORDERCOLOR=#000000 >
```

```
<FONT SIZE=2 FACE="Arial" COLOR=#000000>CategoryID</FONT></TH>
<TH BGCOLOR=#c0c0c0 BORDERCOLOR=#000000 >
<FONT SIZE=2 FACE="Arial" COLOR=#000000>QuantityPerUnit</FONT></TH>
<TH BGCOLOR=#c0c0c0 BORDERCOLOR=#000000 >
<FONT SIZE=2 FACE="Arial" COLOR=#000000>UnitPrice</FONT></TH>
<TH BGCOLOR=#c0c0c0 BORDERCOLOR=#000000 >
<FONT SIZE=2 FACE="Arial" COLOR=#000000>UnitsInStock</FONT></TH>
<TH BGCOLOR=#c0c0c0 BORDERCOLOR=#000000 >
<FONT SIZE=2 FACE="Arial" COLOR=#000000>UnitsOnOrder</FONT></TH>
<TH BGCOLOR=#c0c0c0 BORDERCOLOR=#000000 >
<FONT SIZE=2 FACE="Arial" COLOR=#000000>ReorderLevel</FONT></TH>
<TH BGCOLOR=#c0c0c0 BORDERCOLOR=#000000 >
<FONT SIZE=2 FACE="Arial" COLOR=#000000>Discontinued</FONT></TH>
</TR></THEAD>
<TBODY>
<%
On Error Resume Next
rs.MoveFirst
do while Not rs.eof
%>
<TR VALIGN=TOP>
<TD BORDERCOLOR=#808080 ><B>
<FONT SIZE=1 FACE="MS Sans Serif" COLOR=#000000>
<%=Server.HTMLEncode(rs.Fields("ProductID").Value)%><BR></FONT>
</B></TD>
<TD BORDERCOLOR=#808080 >
<FONT SIZE=1 FACE="MS Sans Serif" COLOR=#000000>
<%=Server.HTMLEncode(rs.Fields("ProductName").Value)%><BR>
</FONT></TD>
<TD BORDERCOLOR=#808080 >
<FONT SIZE=1 FACE="MS Sans Serif" COLOR=#000000>
<%=Server.HTMLEncode(rs.Fields("SupplierID").Value)%><BR>
</FONT></TD>
<TD BORDERCOLOR=#808080 >
<FONT SIZE=1 FACE="MS Sans Serif" COLOR=#000000>
<%=Server.HTMLEncode(rs.Fields("CategoryID").Value)%><BR>
</FONT></TD>
<TD BORDERCOLOR=#808080 >
<FONT SIZE=1 FACE="MS Sans Serif" COLOR=#000000>
<%=Server.HTMLEncode(rs.Fields("QuantityPerUnit").Value)%><BR>
</FONT></TD>
<TD BORDERCOLOR=#808080  ALIGN=RIGHT>
<FONT SIZE=1 FACE="MS Sans Serif" COLOR=#000000>
<%=Server.HTMLEncode(rs.Fields("UnitPrice").Value)%><BR>
```

```
</FONT></TD>
<TD BORDERCOLOR=#808080  ALIGN=RIGHT>
<FONT SIZE=1 FACE="MS Sans Serif" COLOR=#000000>
<%=Server.HTMLEncode(rs.Fields("UnitsInStock").Value)%><BR>
</FONT></TD>
<TD BORDERCOLOR=#808080  ALIGN=RIGHT>
<FONT SIZE=1 FACE="MS Sans Serif" COLOR=#000000>
<%=Server.HTMLEncode(rs.Fields("UnitsOnOrder").Value)%><BR>
</FONT></TD>
<TD BORDERCOLOR=#808080  ALIGN=RIGHT>
<FONT SIZE=1 FACE="MS Sans Serif" COLOR=#000000>
<%=Server.HTMLEncode(rs.Fields("ReorderLevel").Value)%><BR>
</FONT></TD>
<TD BORDERCOLOR=#000000  ALIGN=RIGHT><B>
<FONT SIZE=2 FACE="System" COLOR=#000000>
<%=Server.HTMLEncode(rs.Fields("Discontinued").Value)%><BR>
</FONT></B></TD></TR>
<%
rs.MoveNext
loop
%>
</TBODY><TFOOT></TFOOT>
</TABLE></BODY></HTML>
```

שים לב, שיש בקוד תערובת של HTML ו-VBScript. כמו כן נעשה בו שימוש ב-ADO כדי ליצור את החיבור אל מקור נתונים, לשלוף ערכת רשומות, ולצרף את התוצאות לטבלת HTML. לולאת Do מעבירה את תוכן ערכת הרשומות אל תוך טבלת HTML. שום חלק מהקוד ב-VBScript אינו נשאר בגירסה שהשרת משגר אל הדפדפן ; הדפדפן מקבל HTML טהור. תכונה זאת מתאימה את ASP לסוגים רבים ושונים של דפדפנים. הגמישות שבכתיבת HTML במסגרת העבודה השוטפת היא הגורם להשהיה כאשר דפדפן מפעיל קובץ ASP. אם דפים מסוימים מכילים יותר חזרות ושדות לתרגום, נדרש להם יותר זמן חיבור מאשר לדפים המכילים מספר שדות ושורות בלבד. מומלץ לתכנן את תוכן דפי ה-ASP כך שיכילו כמות מספקת של מידע בדף יחיד, מבלי לגרום לעיכובים מיותרים בעיבוד.

שימוש בטפסי HTML

אחד היתרונות של טכנולוגיית ASP הוא התמיכה שלה בטפסי HTML. נקודה זאת חשובה, הואיל וכל הדפדפנים תומכים בטפסי HTML. תוכל להשתמש בטפסים אלה לאיסוף נתונים ממבקרים באתר, או להחזרת תוצאות מסד נתונים במבנה רשומה (לעומת מבנה גיליון נתונים). טופס HTML כולל פקד אחד, או יותר, להצגה או שמירה של נתונים. ככלל, הטופס מכיל לפחות לחצן אחד למשלוח שדות הטופס אל שרת ה-Web. אם מדובר בטופס המחייב מענה, אחת התוכניות בשרת תוכל לעבד את שדות הטופס, ולהפיק דף תגובה.

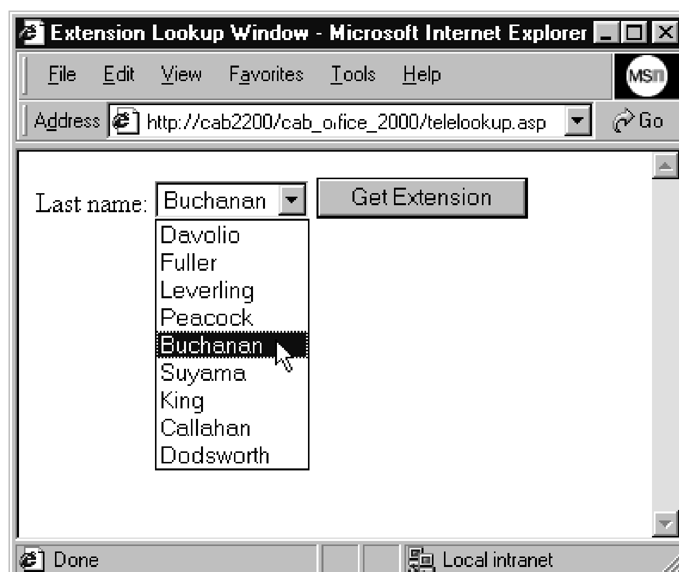
במידה רבה, טפסי HTML פועלים כמשחק קריאה-ותגובה בין דפדפן ושרת Web. המשתמש בצד הדפדפן ממלא את השדות ולוחץ על הלחצן **שלח** (Submit) בטופס. כאשר הדפדפן שולח את הטופס לשרת, הוא מעביר את ערכי השדות בטופס, וגם שם של תוכנית בשרת אשר יודעת מה לעשות באותם ערכים. זהו חלק ה׳קריאה׳ במשחק. שרת ה-Web שולח את ערכי שדות הטופס אל תוכנית העיבוד המתאימה בשרת. תוכנית זאת יכולה ליצור הד של ערכי הקלט, לבדוק את חוקיותם, לצרף אותם למקור רשומות, לבצע בדיקת מידע לנתונים מתוך מקור רשומות, ועוד. בדרך כלל היא מכינה דף החזר כלשהו לדפדפן. זהו חלק ה׳תגובה׳. קבצי ASP יכולים לשמש הן כטופס הראשוני המקבל את הקריאה, והן כתוכנית התגובה המשיבה לקריאה.

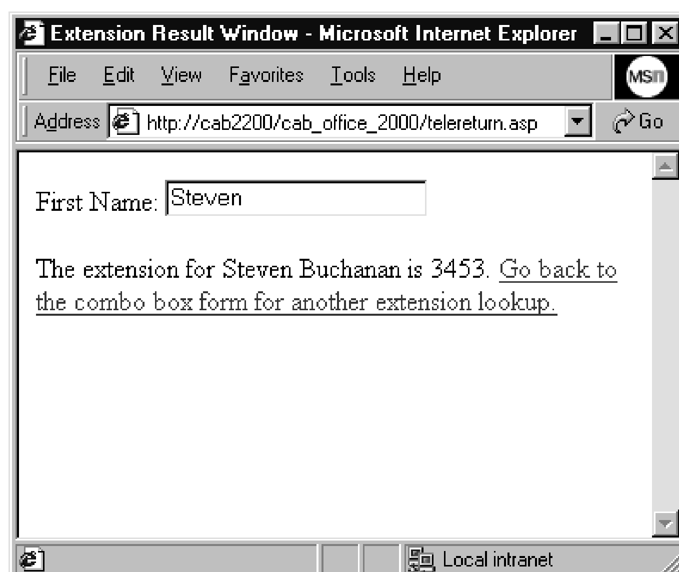
תרשים 13.4 מציג זוג טפסי HTML המדגימים את אופי הקריאה-ותגובה המאפיין את העיבוד של טפסי HTML. הטופס העליון כולל תיבת רשימה נפתחת עם שמות של עובדים במסד הנתונים Northwind. המשתמש יכול לבחור שם, לדוגמה Buchanan, וללחוץ על הלחצן **Get Extention** כדי לשגר את הטופס לשרת. זוהי פעולת הקריאה. שרת ה-Web מעביר את טופס הקלט לקובץ telereturn.asp, הקורא את הערך בתיבת הרשימה, מחפש את מספר הטלפון של העובד המבוקש, וכותב עבור הדפדפן דף המכיל את הנתונים המבוקשים. זוהי פעולת התגובה.

Access 2000 אינו כולל אשפים מוכללים לבניית דפים מסוג זה עם טפסי HTML, אולם התהליך אינו מסובך כלל. האתגר המרכזי הוא למידת התחביר של מספר פקדים לטפסי HTML, וכיצד ליצור בתוך קובץ ASP עירוב בין HTML ו-script (למשל VBScript). דוגמת הקוד הבאה היא כל שנדרש לדף העליון בתרשים 13.4. שם הקובץ של קוד זה הוא telelookup.asp. שם זה מופיע בתיבת הרשימה **Address** שבדף העליון בתרשים.

```
<%
set cnn1 = server.createobject("ADODB.Connection")
cnn1.Open "Provider=Microsoft.Jet.OLEDB.4.0;" & _
   "Data Source=C:\Program Files\Microsoft Office\Office\" & _
   "Samples\Northwind.mdb;"
set rs = Server.CreateObject("ADODB.Recordset")
rs.open "select * from employees", cnn1
%>
<html>
<head><title>Extension Lookup Window</title></head>
<body>
<form name=MyForm method=Post action=telereturn.asp>
Last name: <select name=DCombo>
<%
do while not rs.eof
%>
   <option value=<%=rs.fields("EmployeeID")%>>
      <%=rs.Fields("LastName")%>
<%
rs.MoveNext
Loop
%>
```

```
</select>
<input type="Submit" value="Get Extension"><br><br>
</form></body></html>
```





תרשים 13.4: טופס HTML בתמונה העליונה קורא לשרת ה-Web וגורם ליצירת דף המענה שבתמונה התחתונה

שים לב שה-script תחום בסימנים <% ו- %>. מקטע ה-script הראשון יוצר חיבור אל מסד הנתונים Northwind באמצעות מנהלי OLE DB שאינם זקוקים ל-DSN. המקטע הראשון אף יוצר ערכת רשומות עם הנתונים שיאכלסו את תיבת הרשימה הנפתחת. כעת, חלק מקוד HTML פותח דף, ועליו טופס. בהצהרת הטופס, הקוד מגדיר את telereturn.asp כתוכנית שתעבד את הטופס. ממש לפני תחילתה של לולאת Do, התוכנית מצהירה על תיבת רשימה נפתחת עם מילת המפתח select של שפת HTML. היא מעניקה לפקד את השם dcombo (שמות בשפת HTML אינם תלויי-רישיות). לולאת ה- Do מאכלסת את תיבת הרשימה בערכים מתוך ערכת הרשומות שיצר מקטע ה-script הראשון. הטופס מסתיים בלחצן **Submit** בעל תווית **Get Extention**.

הקובץ telereturn.asp פותח אף הוא ביצירת חיבור וערכת רשומות, כפי שמוצג להלן. אולם פעולת פתיחה זאת נבדלת בשני מובנים חשובים מזו של הקובץ telelookup.asp. ראשית, קובץ זה משתמש ב- DSN כדי לעזור ביצירת החיבור. דבר זה אינו הכרחי, אלא שקבצי ASP רבים מעוצבים באופן זה. שנית, הקוד קורא את הערך של dcombo, ומשתמש בו כדי ליצור את משפט SQL לשאילתה שתחפש את מספר השלוחה של העובד.

```
<%
set rs = Server.CreateObject("ADODB.Recordset")
sql = "select * from employees"
sql = sql & " where employees.employeeid = "
sql = sql & request.form("dcombo")
rs.open sql,"DSN=Nwind2000"
%>
<html>
<head><title>Extension Result Window</title></head>
<body>
<form>
First Name: <input TYPE="TEXT" VALUE="
   <%=rs.fields("FirstName")%>">
</form>
The extension for <%=rs.fields("FirstName")%> 
   <%=rs.fields("LastName")%>
   is <%=rs.fields("Extension")%>.
   <a href="telelookup.asp">
   Go back to the combo box form for another extension lookup.
</a>
</body></html>
```

לאחר שיצר ערכת רשומות עם הנתונים המבוקשים עבור העובד שנבחר בקובץ telelookup.asp, הקובץ telereturn.asp מעצב דף. הדף מכיל את שמו הפרטי של העובד בתיבת טקסט HTML, מחרוזת HTML פשוטה עם השם ומספר השלוחה, והיפר-קישור להחזרת השליטה לקובץ telelookup.asp. כאשר שרת ה- Web מעביר את הדף בחזרה לדפדפן, מסתיים מחזור הקריאה-ותגובה.

שימוש בתמונות FTP עם דפדפני Netscape

פרק 6 עסק בתבנית התמונה (snapshot) לדוחות, והציג דוגמאות של יכולות התבנית, אולם לא הזכיר שימוש בתמונות עם FTP. כאשר מציבים תמונות בתיקיית FTP המיועדת לשרת Web, דפדפני Netscape יכולים לפתוח אותן מרחוק בשרת ה-Web. המשתמש אינו חייב לטעון את הקובץ אל המחשב שלו ולהפעיל מציג נפרד, כפי שנדרש בפרוטוקול HTTP.

כאשר דפדפן Netscape פותח תמונה של דוח מתוך תיקיית FTP, הוא יוצר באופן אוטומטי חלון חדש עבור אותו דוח. תרחיש זה מניח ש- **Snapshot Viewer** (מציג התמונות) מותקן במחשב שלך, וכי יש בתחנת העבודה שיוך בין קבצים מסוג snp.* לבין המציג. תרשים 13.5 מציג מקטע מתוך תמונה של הדוח Catalog במסד הנתונים Northwind. התרשים מציג ציור, גופנים מרובים וסממני עיצוב נוספים, כמו למשל קווים מפרידים. הטופס להזנת הזמנות שבסוף הדוח Catalog אף הוא העתק נאמן.



Product Name:	Product ID:	Quantity Per Unit:	Unit Price:
Chai	1	10 boxes x 20 bags	$18.00
Chang	2	24 - 12 oz bottles	$19.00
Chartreuse verte	39	750 cc per bottle	$18.00
Côte de Blaye	38	12 - 75 cl bottles	$263.50
Guaraná Fantástica	24	12 - 355 ml cans	$4.50
Ipoh Coffee	43	16 - 500 g tins	$46.00
Lakkalikööri	76	500 ml	$18.00
Laughing Lumberjack Lager	67	24 - 12 oz bottles	$14.00
Outback Lager	70	24 - 355 ml bottles	$15.00
Rhönbräu Klosterbier	75	24 - 0.5 l bottles	$7.75
Sasquatch Ale	34	24 - 12 oz bottles	$14.00
Steeleye Stout	35	24 - 12 oz bottles	$18.00

תרשים 13.5: מקטע מתוך תמונת דוח, שנפתחה בעזרת דפדפן Netscape

בעת שמירת קובץ התמונה, הצב אותו בתיקיה FTPROOT עבור שרת ה-Web (או בתיקיית FTP אחרת). בתיבת הכתובת **Location** ב-Netscape, הקלד /ftp://webserver filename.snp. הצב את שם שרת ה-Web שלך במקום webserver (בדוגמאות שבפרק זה, שרת ה- Web הוא cab2200). מכיון שפרוטוקול FTP נפוץ ומקובל מאוד, תוכל להיעזר במנגנון פרסום זה לעבודה עם סוגים רבים של שרתי Web ודפדפני Web.

שימוש בהיפר-קישורים

בפרק 5 למדנו כיצד להשתמש בהיפר-קישורים כדי לנווט בטפסים בתוך יישום מותאם אישית. זהו רק חלק קטן מהפונקציונליות הטמונה בהיפר-קישורים. תוכל לנווט גם אל מסדי נתונים אחרים של Access, אל סימניות במסמך Word, או אל כתובות URL ברשת ארגונית ובאינטרנט. תוכל להשתמש בהיפר-קישורים גם להפעלת תוכנת הדואר האלקטרוני שלך (כמו למשל Microsoft Outlook) עם כתובת של פרט מוגדר ונושא נתון.

סוגים של היפר-קישורים

היפר-קישור יכול להופיע כתווית, כלחצן פקודה, או כפקדי תמונה לא מאוגדים בטפסים, בדוחות ובדפי גישה לנתונים. כמו כן ניתן ליצור היפר-קישורים מאוגדים, כך שטבלה תוכל להכיל סדרה של היפר-קישורים. באפשרותך להקצות היפר-קישור ייחודי לכל רשומה בטבלה. אחד השימושים הברורים מאליהם באפשרות זאת הוא שיוך כתובת URL לכל ישות רשומה (כמו למשל, חברות). לאחר שעקבת אחר היפר-קישור מאוגד או בלתי מאוגד, תוכל להשתמש בפקד **Back** (הקודם) בסרגל הכלים של הדפדפן או בסרגל הכלים **אינטרנט** (Web) של Office, כדי לנווט בחזרה אל Access מכתובת יעד של היפר-קישור.

סוג הנתונים Hyperlink

כאשר משתמשים בהיפר-קישורים מאוגדים, היישום מציין שסוג הנתונים הוא **היפר-קישור** (Hyperlink). סוג נתונים זה עשוי להכיל עד ארבעה חלקים, שכל אחד מהם מופרד באמצעות הסימן #. החלק של טקסט התצוגה יופיע ראשון אם תציין זאת. טקסט זה מופיע במקום הכתובת שאליה מוביל ההיפר-קישור. לדוגמה, במקום היפר-קישור המראה את הכתובת של אתר Microsoft ב- Web, ניתן להציג את הכיתוב Microsoft.

החלק השני מכיל את הכתובות – כתובת URL או UNC של קובץ היעד. הכתובת עשויה להיות מוחלטת או יחסית. באתרי Web, מקובל לציין היפר-קישורים באופן יחסי. ב-Access, כתובת יחסית נגזרת מההגדרה **בסיס היפר-קישור** (Hyperlink Base) בקובץ Access. תוכל לבדוק הגדרה זאת על ידי בחירה בפקודה **מאפייני מסד נתונים** (Database Properties) מתפריט **קובץ** (File) שבחלון מסד הנתונים.

החלק השלישי הוא הכתובת המשנית, המכילה את שם החלק של הקובץ שיש לנווט אליו. החלקים שניתן לנווט אליהם תלויים ברכיב Office. ב-Access ניתן לנווט אל אובייקטים של מסד נתונים, כמו למשל טפסים, דוחות, טבלאות ושאילתות. תוכל לציין סוג של אובייקט מסד נתונים, כמו למשל טבלה, ואת שמו של אובייקט מוגדר, כמו למשל Customers. אינך נדרש לציין את סוג האובייקט, אולם אם מסד נתונים מכיל מספר אובייקטים באותו שם, Access ינווט אליהם בסדר אקראי. אם היפר-קישור מכיל הן כתובת והן כתובת משנה, הקישור ינווט אל קובץ אחר. אם חסר החלק המכיל את הכתובת, הקישור ינווט אל אובייקט במסד הנתונים הנוכחי.

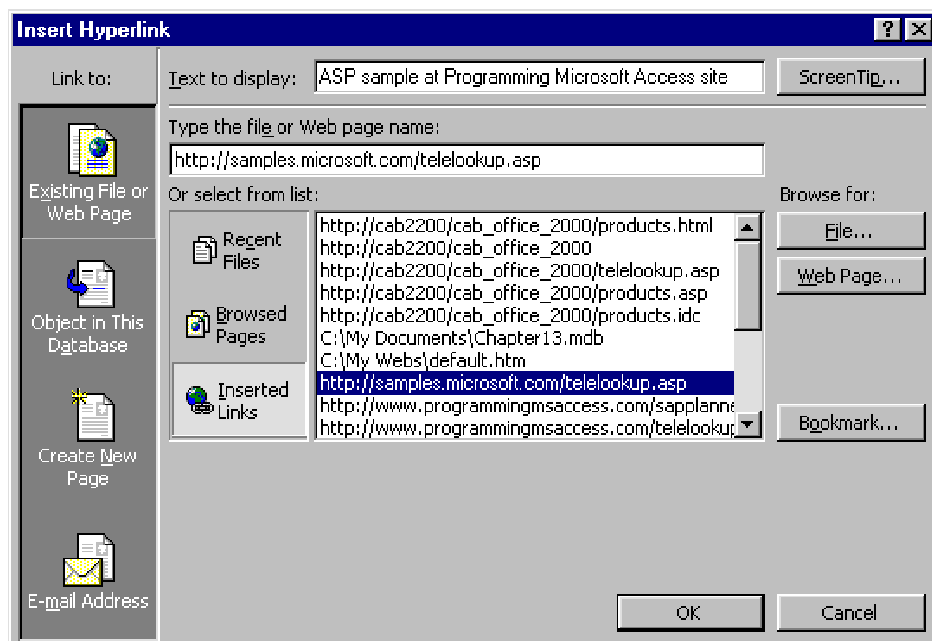
החלק הרביעי, תיאור המסך, חדש ב- Access 2000. הטקסט מופיע כאשר מניחים את הסמן על היפר-קישור. הוא עשוי להכיל תזכורת אודות מטרתו של היפר-קישור. תיאורים אלה מופיעים עבור היפר-קישורים בדפים מוצגים בעזרת Internet Explorer 4 ומעלה. כמו כן הם מופיעים ביישומים מותאמים אישית ב-Access 2000.

הוספה ועריכה של היפר-קישורים

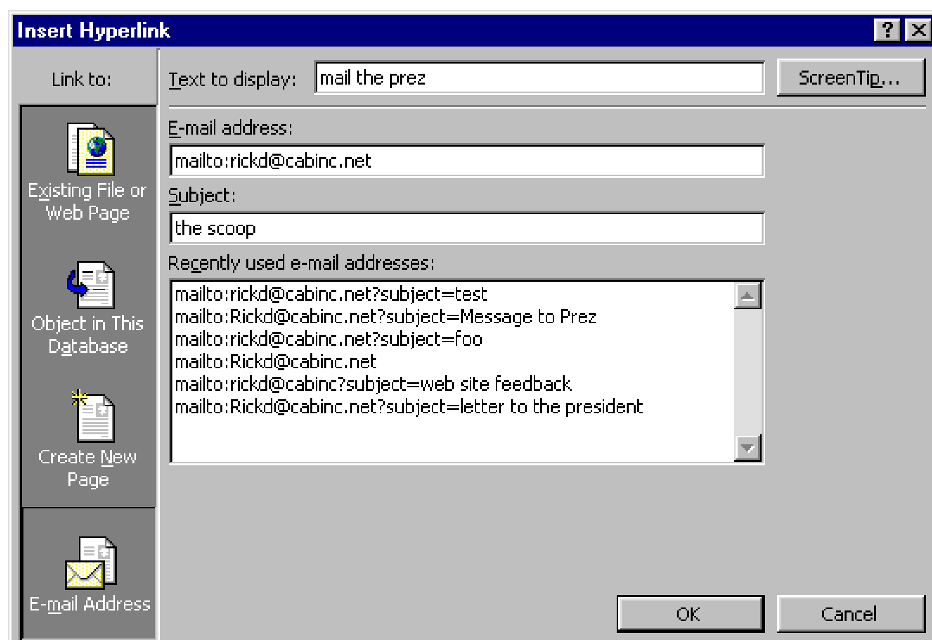
כאשר מקלידים כתובת היפר-קישור חדשה, פשוט ובטוח יותר להשתמש בכלי **הוסף היפר-קישור** (Insert Hyperlink), או בפקודת התפריט המקבילה. תוכל להפעיל את הפקודה **הוסף היפר-קישור** מכל נקודה שבה אפשר ליצור היפר-קישור במסמך. לדוגמה, תוכל לבחור פקודה זאת מתוך התצוגה **עיצוב** (Design) של טופס, אם תרצה להוסיף היפר-קישור כתווית. תוכל להעניק ללחצן פקודה יכולות ניווט על ידי לחיצה על הלחצן **בנייה** (Build) הסמוך למאפיין **כתובת_היפר-קישור** (Hyperlink Address) או **כתובת משנה של היפר-קישור** (Hyperlink SubAddress) של לחצן הפקודה. בדרך דומה, תוכל להעניק יכולות ניווט גם לפקד ציור.

כל הפעולות הללו פותחות את תיבת הדו-שיח **הוספת היפר-קישור** בטופס. ערך הכיתוב של התווית הוא דוגמת ASP שבאתר הדמיוני Programming Microsoft Access, שכתובת ה- URL שלו היא http://samples.microsoft.com/telelookup.com. לחץ על **תיאור מסך** (ScreenTip) כדי לפתוח תיבת דו-שיח שנייה להזנת טקסט עבור תיאור המסך המשויך לאותו היפר-קישור.

תוכל ללחוץ על הלחצן **כתובת דואר אלקטרוני** (E-Mail Address) בפס האפשרויות כדי לפתוח את תיבת הדו-שיח בגירסה מתאימה לדואר אלקטרוני (ראה תרשים 13.7). תיבת הדו-שיח מוסיפה באופן אוטומטי קידומת פרוטוקול mailto כדי להפעיל מהדפדפן את תוכנת ברירת המחדל לדואר אלקטרוני בתחנת עבודה. כמו כן תוכל לציין את הנמען ואת הנושא. בדומה לבחירה מתוך אתרים שכבר ביקרת בהם בעבר בעת שאתה מקצה כתובת היפר-קישור, תוכל לבחור מבין כתובות דואר אלקטרוני ונושאים שכבר הזנת בעבר. ההיפר-קישור שתרשים 13.7 יוצר מציג את המשפט mail the prez. כאשר משתמש לוחץ על משפט זה, הוא מפעיל את תוכנת הדואר האלקטרוני של המשתמש, כאשר הנמען הוא rickd@cabinc.net, והנושא הוא the scoop.



תרשים 13.6: תיבת הדו-שיח **הוספת היפר-קישור** של כתובת URL



תרשים 13.7: תיבת הדו-שיח **הוספת היפר-קישור** להודעה בדואר אלקטרוני

לאחר הוספת היפר-קישור ביישום, תוכל ללחוץ עליו לחיצה ימנית בעכבר ולבחור באפשרות **היפר-קישור** מתפריט הקיצור כדי לראות את האפשרויות לעיבוד היפר-קישורים קיימים. הפקודה **עריכת היפר-קישור** (Edit Hyperlink) היא הבחירה הטבעית לצורך הכנסת שינויים בהיפר-קישור קיים. הפקודה פותחת את תיבת הדו-שיח **עריכת היפר-קישור**, שעיצובה זהה לתיבת הדו-שיח **הוספת היפר-קישור**. השתמש בתיבה זאת לעדכון השדות הרצויים בהיפר-קישור. לאורך זמן, ייתכן שתצטרך ללטש את הטקסט המוצג או את כתובת ה- URL של היפר-קישור. בדומה לכך, תוכל להוסיף ערכים ברשימת הנמענים של קישורים המשגרים דואר אלקטרוני, כדי שהודעות ינותבו אל יותר נמענים באופן ישיר.

דוגמאות של היפר-קישורים

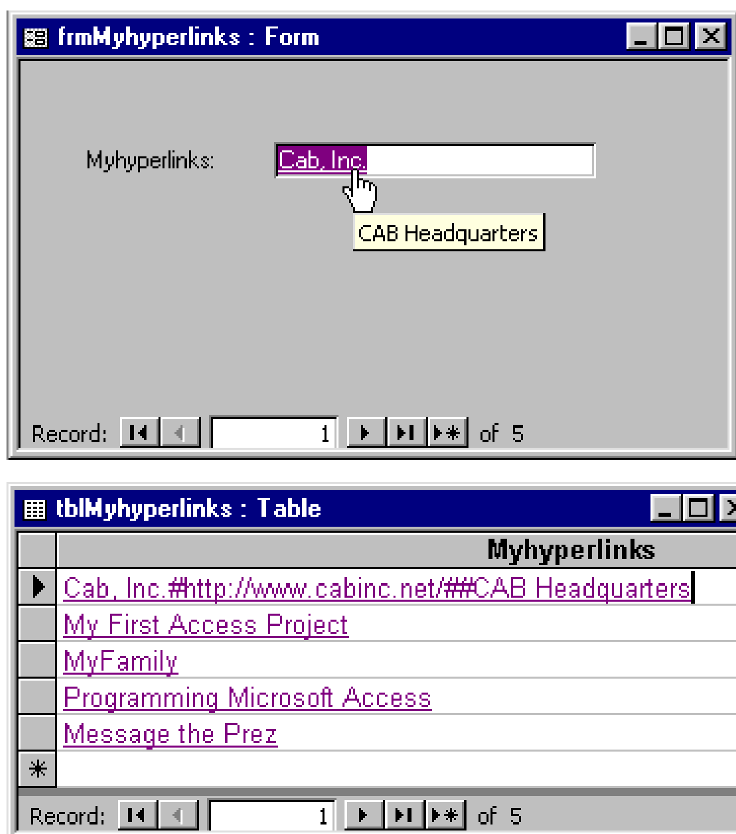
אחד הדפים הפופולריים ביותר באתרי Web רבים הוא דף מועדפים (Favorites) שהוא טופס Access המצביע אל אתרים בעלי ערך או מעניינים בעיני המחבר. באפשרותך להוסיף דפים דומים בפתרונות Access מותאמים אישית. דף מועדפים יכול להכיל אתרים כמו קישורי דואר אלקטרוני או קישורי Web אל בית העסק שלך, וכן כתובות של אתרים המכילים תמיכה טכנית במחיר נמוך או בחינם.

תרשים 13.8 מציג דף מועדפים במתכונת של טופס Access, עם שני ציורים ושתי תוויות. כל הארבעה משמשים כהיפר-קישורים. הציור הראשון הוא לוגו החברה שלי. כאשר משתמש עם חיבור פעיל לאינטרנט לוחץ על הציור, Access מוביל אותו לאתר ה-Web של החברה. הציור השני הוא הלוגו של Microsoft Access, שמוביל את המשתמש אל אתר Access בתוך אתר Microsoft. שני ההיפר-קישורים במתכונת תווית פותחים קישורי דואר אלקטרוני אל עובדים בחברה.



תרשים 13.8: דף מועדפים המכיל שני ציורים ושתי תוויות, המשמשים כולם כהיפר-קישורים

תרשים 13.9 מציג טופס עם היפר-קישור מאוגד המבוסס על טבלה של היפר-קישורים. הטבלה מופיעה בתמונה שמתחת לטופס בתרשים. תיאור המסך, CAB Headquarters, מופיע עבור ההיפר-קישור הראשון. התצוגה התחתונה מראה את ההיפר-קישור האמור במצב עריכה, החושף את חלקי ההיפר-קישור. שים לב שיש טקסט תצוגה וכן חלק המכיל כתובת, אך אין כתובת משנה. ההיפר-קישור מסתיים בטקסט תיאור המסך.



תרשים 13.9: טופס עם היפר-קישור מאוגד המציג תיאור מסך עבור ההיפר-קישור הראשון בטבלה; הטבלה חושפת את התחביר המלא של ההיפר-קישור הראשון

יצירת דפי גישה לנתונים והשימוש בהם

דפי גישה לנתונים הם קבצי HTML מאוגדי-נתונים. במובנים רבים, הם נראים ומתנהגים כמו הכלאה בין דוח Access לטופס Access, מכיון שניתן להשתמש בהם לדפדוף בין הרשומות במקור נתונים. החיבור של הדף למסד נתונים מאפשר למשתמש להוסיף, לעדכן או למחוק נתונים. בדרך כלל, אולם לא בהכרח, הנתונים באים מתוך אובייקט מסד נתונים בקובץ מסד הנתונים הנוכחי. תוכל להשתמש בכלי **מיון וקיבוץ**

(Sorting and Grouping) להצגת נתונים באופן היררכי – לדוגמה, בדוחות עם כותרות קבוצות או מקטעים מרובים ומקוננים. הסמל **עמודים** (Pages) שבחלון מסד הנתונים מייצג את אובייקטי דפי הגישה לנתונים המשויכים למסד נתונים. אולם Access אינו שומר דפים במסד הנתונים, אלא כקבצי DHTML במערכת הקבצים, לדוגמה, בתיקיה בשרת הרשת הארגונית. האוסף **עמודים** (Pages) שנמצא בקובץ מסד נתונים הוא אוסף של היפר-קישורים אל קבצי ה- DHTML. יצירת הדפים נעשית במסגרת Access, ואילו המשתמשים יכולים לפתוח אותם ב- Internet Explorer 5 או ב- Access 2000.



הערה:

אם משתמשים מרובים יפעילו דף באמצעות דפדפנים בתחנות עבודה שונות, ייתכן שתיאלץ לשנות את ההגדרה **Data Source** בקבצי ה-DHTML, כך שלא תפנה אל אותיות של כוננים מקומיים. השתמש בכתובת UNC (Uniform Naming Convention) במקום זאת.

יצירת דף גישה לנתונים

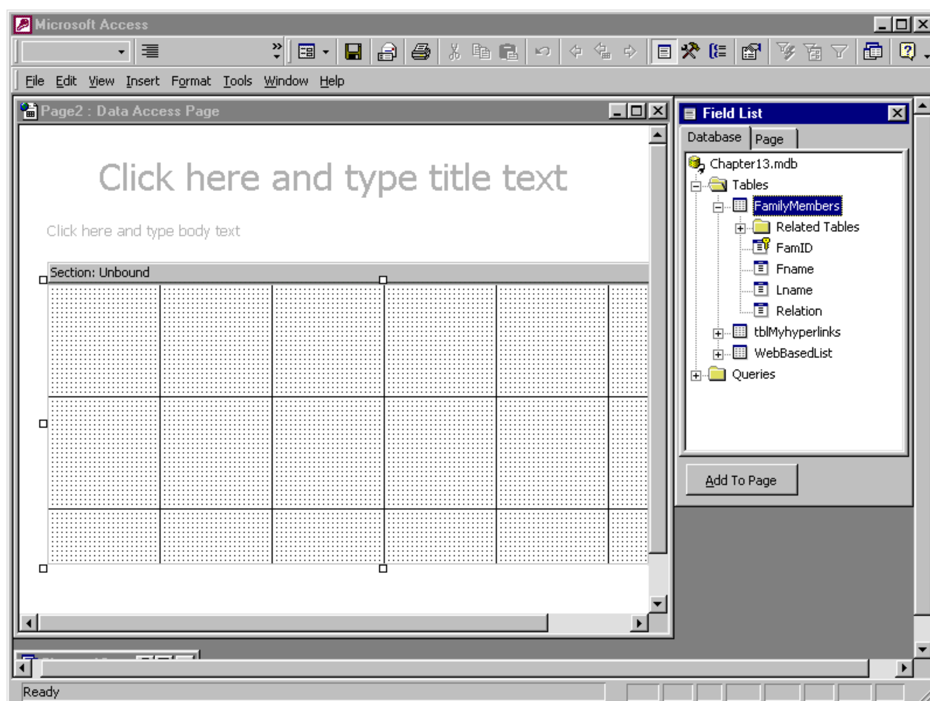
Access 2000 מכיל אשפים ליצירת דפי גישה לנתונים על בסיס טבלאי או במתכונת של עמודות. אפשר גם לבסס דף חדש על דף Web קיים. עם זאת, רבים יעדיפו ליצור דף גישה לנתונים על ידי פתיחת דף ריק בתצוגת **עיצוב** (Design) ואכלוסו בפקדים מתוך תיבת הדו-שיח **רשימת שדות** (Field List) (המוצגת בתרשים 13.10).

כדי לאכלס את הדף, תוכל לפתוח את הצומת **טבלאות** (Tables) או **שאילתות** (Queries) ולחשוף את שמות הטבלאות בחיבור הפעיל. אם תיתקל בטבלה שתרצה לבחור ממנה ערכים לדף החדש, תוכל להרחיב את הטבלה ולסקור את השדות הנפרדים. בשלב זה תוכל לבחור שדות על ידי גרירתם ושחרורם בדף החדש. לחילופין, תוכל לאכלס את הדף בכל השדות של טבלה נתונה על ידי גרירת שם הטבלה אל הדף. לאחר מכן תוכל לסדר את השדות בעיצוב של טבלת עמודות או טבלת ציר. אותן פעולות תוכל לבצע גם בשאילתות.



הערה:

העיצוב של דוח טבלת ציר (Pivot Table שמקורו ב- Excel), הוא עיצוב טבלאי מיוחד שמפשט יצירה של סיכומים מקובצים ואינטראקטיביים של הנתונים במקור רשומות. נראה שעתידן של טבלאות ציר כממשק חזית פופולרי לעבודה עם קוביות נתונים רב-מימדיות, מובטח.

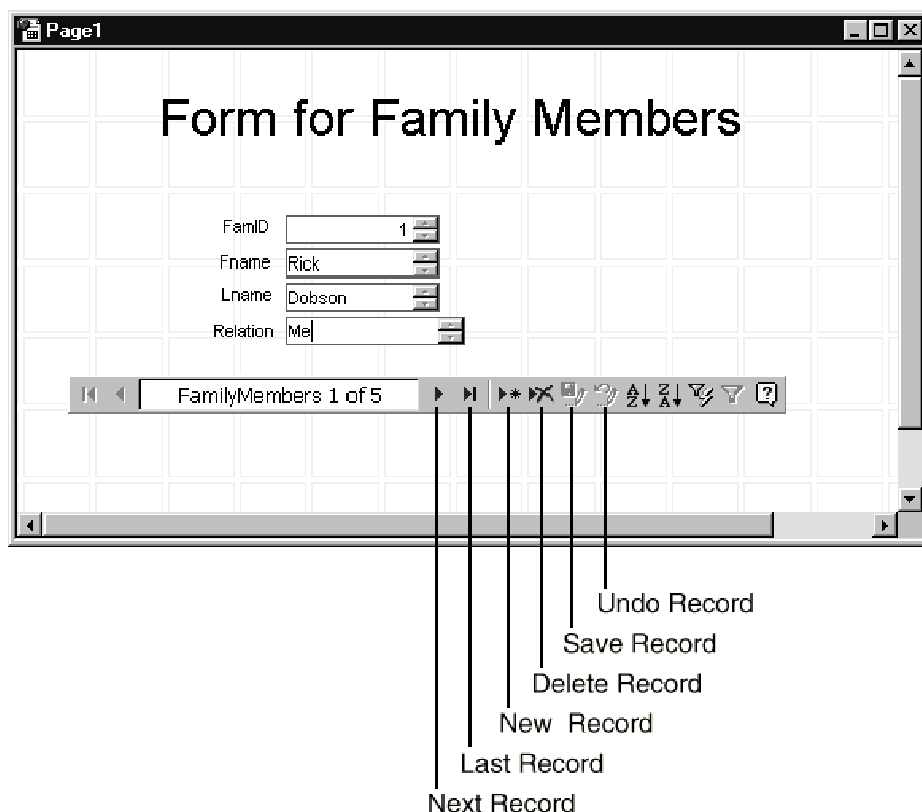


תרשים 13.10: השתמש בתיבת הדו-שיח **רשימת שדות** כדי לאכלס דף גישה לנתונים ריק בפקדים המאוגדים אל שדות של מסד נתונים

כדי לשנות את החיבור הפעיל, לחץ לחיצת עכבר ימנית על מחבר מסד הנתונים בפינה השמאלית-העליונה של תיבת הדו-שיח **רשימת שדות**, ובחר באפשרות **התקשרות** (Connection) מתוך תפריט הקיצור. בתיבת הדו-שיח **Data Link Properties** (מאפייני קישור נתונים) שתופיע על המסך, בחר במנהל חיבור OLE DB ובמסד נתונים המתאים למנהל התקן זה. כאשר תסגור את תיבת הדו-שיח, תציג תיבת הדו-שיח **רשימת שדות** את הטבלאות והשאילתות ממקור מסד הנתונים החדש.

יצירת דף טורי פשוט והשימוש בו

תרשים 13.11 מציג דף גישה לנתונים פשוט המשמש כטופס עבור הטבלה FamilyMembers שנמצאת בקובץ מסד נתונים. הטופס מציג ארבעה שדות. הפקד שמתחת לשדות תומך בניווט ובפונקציות נוספות. לחיצה על שני לחצני הניווט בשני צידי חלון מציין הרשומות תגרום למעבר קדימה ואחורה בין הרשומות בערכת הרשומות שעליה מבוסס הדף.



תרשים 13.11: דף גישה לנתונים המשמש כטופס; המשתמש יכול לדפדף, להוסיף, למחוק, לערוך, למיין ולסנן רשומות בעזרת פקד הניווט שמתחת לשדות

ששת הלחצנים הראשונים מימין למציין מקור ומספר הרשומה הם:

- לחצן Next Record (הרשומה הבאה). מעבר לרשומה הבאה.
- לחצן Last Record (הרשומה האחרונה). מעבר לרשומה האחרונה.
- לחצן New Record (רשומה חדשה). הוספת רשומה חדשה.
- לחצן Delete Record (מחיקת רשומה). מחיקת הרשומה הנוכחית.
- לחצן Save Record (שמירת רשומה). שמירת השינויים שנערכו ברשומה הנוכחית.
- לחצן Undo Record (ביטול שינויים). ביטול כל השינויים שבוצעו ברשומה הנוכחית ולא נשמרו.

אם הזנת רשומה שאינך זקוק לה עוד, מחק אותה בעזרת הלחצן Delete Record. כמו כן תוכל לערוך רשומות על ידי הקלדה על גבי התוכן שהן מכילות, הוספת תכנים או מחיקת תכנים קיימים. כדי לשמור שינויים שביצעת, השתמש בלחצן Save Record. אם אינך מעוניין לשמור אותם, לחץ על Undo Record.

שאר הלחצנים בפקד הניווט (להוציא לחצן העזרה שבקצה הימני) מיועדים למיון וסינון:

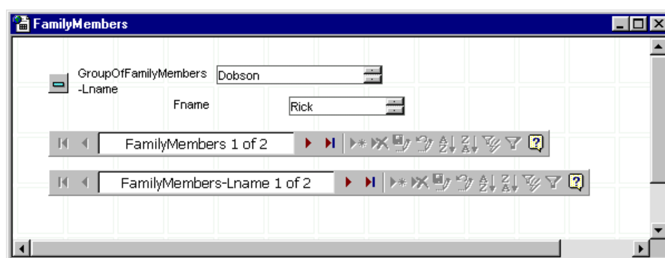
- לחצני **Sort Ascending** (מיון בסדר עולה) ו-**Sort Descending** (מיון בסדר יורד). ציין את כיוון המיון. כדי למיין לפי ערכים בשדה, לחץ בשדה זה ברשומה כלשהי, ולאחר מכן לחץ על אחד מלחצני המיון.
- לחצן **Filter by Selection** (סינון לפי בחירה). מחיל מסנן המבוסס על הערך הנוכחי בשדה. לחץ בתוך השדה כדי לבחור ערך לסינון (לדוגמה, השדה Lname עם הערך Dobson), ולאחר מכן לחץ על הלחצן **Filter by Selection**. בעקבות פעולה זאת התצוגה תשתנה ותציג רק את הרשומות המתאימות לערך הסינון שנבחר. לדוגמה, אם בחרת בערך Dobson עבור השדה Lname בתרשים 13.11, מציין הרשומות ישתנה מ- FamilyMembers 1 of 5 (FamilyMembers 1 מתוך 5) ל- FamilyMembers 1 of 2.
- לחצן Toggle Filter (סינון דו-מצבי). מחיל או מבטל את החלת המסנן.

כל התפקודיות הזאת עומדת לרשות המשתמש מתוך Access עצמו, כמו גם מתוך דפדפן שנמצא ברשת ארגונית (אינטרא-נט). התצוגה המוצלחת ביותר של דפי גישה לנתונים, כולל תפקודיות של כל התכונות, היא באמצעות Internet Explorer 5. ירידה בתפקודיות בדרגות שונות מלווה את השימוש ב- Internet Explorer 4. משתמשים המפעילים דפדפנים אחרים מלבד Internet Explorer 5 ייאלצו להשיג גם פקדי ActiveX כדי לקרוא דפי גישה לנתונים ולטפל בהם.

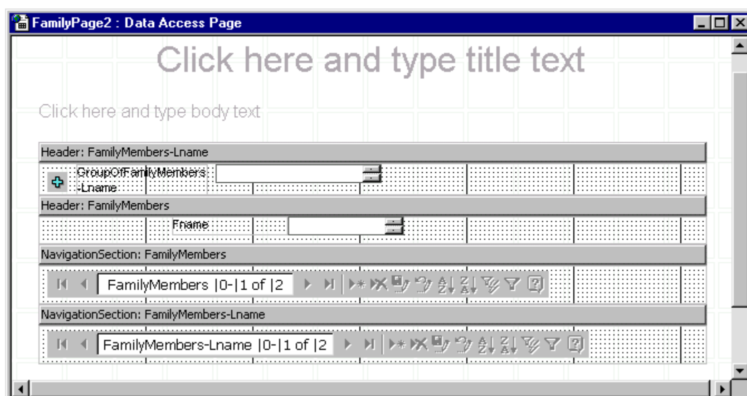
קיבוץ רשומות

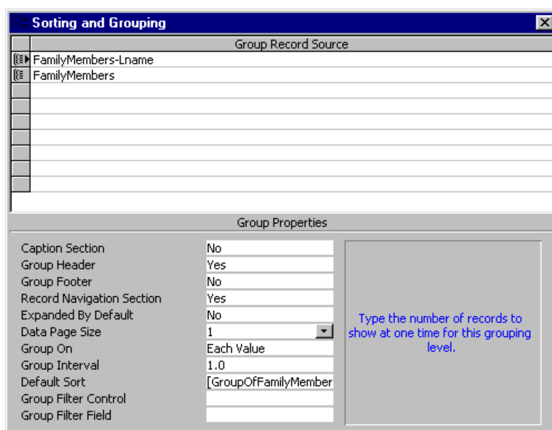
אחת התכונות החשובות של דפי גישה לנתונים היא יכולתם לקבץ רשומות ולהרחיב קבוצה באופן מותנה, כך שהערכים הנפרדים שהיא מכילה ייראו. תרשים 13.12 מציג יכולת זו. הדף הנראה בתרשים מקבץ רשומות מהטבלה FamilyMembers לפי השדה Lname. כל רשומות הטבלה ששם המשפחה בהן זהה, מתקבצות לקבוצה אחת. כאשר המשתמש לוחץ על לחצן ההרחבה האפור (הוא משתנה מ- [+] ל- [–]), מופיע פקד ניווט שני, מקונן, ומאפשר דפדוף בין הרשומות בעלות אותו שם משפחה. לחיצה על פקד הניווט החיצוני תגרום לסגירת התצוגה המורחבת ותעבור אל הרשומה הראשונה עם שם המשפחה הבא.

ניתן להציג יותר מרשומה אחת בדף. השתמש בהגדרה **גודל דף נתונים** (Data Page Size) שבחלון **מיון וקיבוץ** (Sorting and Grouping) כדי לציין כמה רשומות להציג בכל פעם (ראה חלון תחתון בתרשים 13.13). החלון מאפשר לקבוע את גודל הדף באופן בלתי תלוי עבור שדה הקיבוץ כמו גם עבור השדות שבתוך קבוצה. השתמש בחלון **מיון וקיבוץ** לדפי גישה לנתונים כדי לקבוע אם יהיו לקבוצות מקטעי כותרת עליונה וכותרת תחתונה, כפי שאתה משתמש בחלון המקביל כשמדובר בדוחות. השתמש בהגדרה **קיבוץ לפי** (Group By) כדי לציין מרווחים, טווחי ערכים וקידומות לשדות קיבוץ.



תרשים 13.12: ניתן לקבץ ולבצע הרחבה מותנית של רשומות בקבוצות, בדפי גישה לנתונים





תרשים 13.13: בתצוגת **עיצוב**, תוכל להוסיף פקדי הרחבה כדי לאפשר למשתמשים לפעול עם הדפים, ולבצע בהם הרחבה מותנית כדי לראות את הרשומות בתוך קיבוץ נתונה

החלון העליון בתרשים 13.13 מראה את התצוגה **עיצוב** (Design) של הדף מתרשים 13.12. שים לב, שהוא מכיל שני פקדי ניווט. באפשרותך להקצות פונקציית קיבוץ לשדה על ידי בחירת השדה ולחיצה על הלחצן **קדם** (Promote) (החץ המצביע שמאלה) בסרגל הכלים **עיצוב עמוד** (Page Design). כאשר מקור הרשומות של דף נגזר מקשר

גומלין של יחיד-לרבים, ניתן להשתמש בכלי **מיון וקיבוץ** כדי לקבץ את כל השדות מצד ה'יחיד' של קשר הגומלין במקום לפי השדות בצד ה'רבים'. במקרה זה, Access ממיין ומקבץ את הרשומות לפי המפתח הראשי של צד ה'יחיד' בקשר הגומלין.

רכיבי Web של Office 2000 בדפי גישה לנתונים

Office 2000 משווק בלוויית שלושה רכיבי Web (במהדורות Standard, Professional, Premium ו- Developer): גיליון אלקטרוני, טבלה ורשימת טבלת ציר. השתמש ברכיבים אלה להשלמה והרחבה של תפקודיות מסד הנתונים הכלולה בדפי גישה לנתונים. אם באתר מסוים יש רשיון המתיר הפצה ברשת ארגונית, מנהל האתר יוכל להגדיר את תצורת הדפדפנים במסגרת הרשיון כך שיטענו באופן אוטומטי ויגדירו את תצורת רכיבי ה- Web של Office בפעם הראשונה שמזדמן להם לטעון דף בעזרת רכיב (תמצא מידע נוסף על קביעת התצורה של רכיבי Web של Office ב- Microsoft Office 2000 Resource Kit).

דוגמת גיליון אלקטרוני של Office

תרשים 13.14 מציג את אחד השימושים של רכיב הגיליון האלקטרוני בדף גישה לנתונים. הפקדים בדף מציגים את השדות CategoryName, ProductName ו-ProductSales של השאילתה Sales By Category במסד הנתונים Northwind. שאילתה זאת מחשבת את סך-כל המכירות לפי מוצר בשנת 1997. השאילתה מציגה את זיהוי הקטגוריה והשם עבור כל מוצר, לצד נתוני המכירות.

הדוגמה בתרשים 13.14 מרחיבה את השאילתה הבסיסית על ידי יצירת תחזית המכירות משנת 1998 ועד שנת 2001, כולל. תחילה, הרכיב מעתיק את הערך הנוכחי של הפקד ProductSales מהדף אל הגיליון האלקטרוני. לאחר מכן הוא מחשב סדרה הדרגתית של שיעורי הגידול במכירות החל משנת 1997. שיעור זה מגדיל את המכירות משנה לשנה. למרות ששיעורי הגידול זהים לכל המוצרים, רמת המכירות בפועל שונה ממוצר למוצר הואיל ורמת המכירות בפועל ב- 1997 אינה זהה לכל המוצרים. לבסוף, כדי להגן על הנוסחאות מפני נזק, ניתן לנעול גליונות אלקטרוניים נבחרים כך שהמשתמשים לא יוכלו לשנותם בשוגג.

תרשים 13.15 מראה את התצוגה **עיצוב** של דף הגישה לנתונים שבתרשים 13.14. שים לב, שמתחת לפקד ProductSales מופיע גיליון אלקטרוני. באפשרותך להוסיף רכיב גיליון אלקטרוני של Office 2000 בעזרת הפקודות **הוספה** (Insert), **גיליון אלקטרוני של Office** (Office Spreadsheet), ולהתאים את הרכיב לשימוש בתוך היישום שלך על ידי לחיצה ימנית על הרכיב ובחירה באפשרות **ארגז כלים של מאפיינים** (Property Toolbox). בגיליון האלקטרוני שבתרשים 13.15 מוסתרים סרגל הכלים, פס הכותרת, כותרות העמודות וכותרות השורות. בנוסף, ניתן ערך False להגדרות פס הגלילה האופקי והאנכי של הרכיב.

Page1

Projections for Sales by Category Query

from Northwind Database

CategoryName Meat/Poultry

ProductName Alice Mutton

ProductSales $17,604.60

1997	$17,604.60
1998	$19,365.06
1999	$21,785.69
2000	$25,053.55
2001	$29,437.92

1 of 77

תרשים 13.14: רכיב הגיליון האלקטרוני בדף הגישה לנתונים משתמש בנתוני מכירה לשנת 1997 מתוך מסד הנתונים כדי לגבש תחזית מכירות עד 2001, כולל

SalesGrowth : Data Access Page

Projections for Sales by Category Query

from Northwind Database

Header: Sales by Category

CategoryName

ProductName

ProductSales

1997	=document.productsales.value
1998	#VALUE!
1999	#VALUE!
2000	#VALUE!
2001	#VALUE!

NavigationSection: Sales by Category

|0 of |2

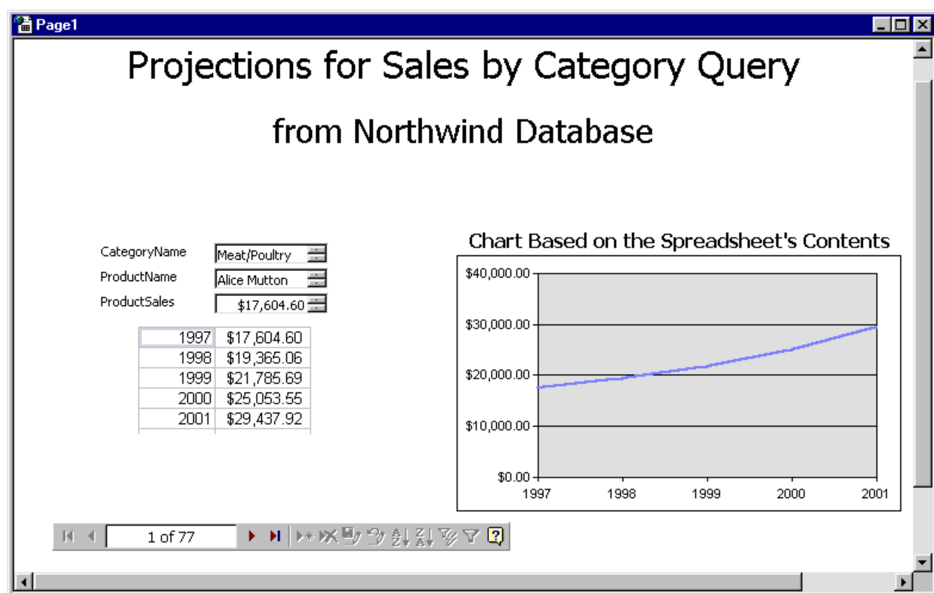
תרשים 13.15: תצוגת עיצוב של דף הגישה לנתונים מתרשים 13.14

תא נתוני המכירות העליון בגיליון האלקטרוני מכיל נוסחה המתעדכנת עם כל רשומה חדשה בדף הגישה לנתונים: document.productsales.value=. האיבר document מפנה לדף גישה לנתונים. האיבר Productsales מפנה לפקד מסוים בדף. ולבסוף, המאפיין value מפנה אל הערך הנוכחי של הפקד בדף הגישה לנתונים. ערך זה משתנה רק אם המשתמש עובר לרשומה חדשה במקור הרשומות שברקע הדף.

רישומי המכירות לשנים 1998 עד 2001, כולל, מחושבים כ- !VALUE# בתצוגה **עיצוב**. זכור, שלפי תרשים 13.14 יש לתאים אלה ערכים חוקיים בתצוגת **עמוד** (Page). דבר זה נובע מהנוסחה שבתא העליון של הגיליון האלקטרוני המשתמשת בערך הנוכחי של productsales בדף הגישה לנתונים.

דוגמת תרשים של Office

הדוגמה המסיימת פרק זה בנויה על הדוגמה הקודמת, ועיקרה תוספת רכיב תרשים של Office 2000. רכיב זה מציג באמצעות תרשים את הערכים שבגיליון האלקטרוני. התרשים מציג באופן גרפי את הגידול במכירות לאורך זמן לכל מוצר. תרשים 13.16 מציג מוצר מסוים, בלוויית התחזיות של הגיליון האלקטרוני ותיאור גרפי של הנתונים בצידו הימני של הדף. התרשים מתעדכן באופן דינמי כל פעם שהערכים בגיליון האלקטרוני משתנים.



תרשים 13.16: רכיב התרשים של Office 2000 בדף גישה לנתונים מקבל ערכים מרכיב גיליון אלקטרוני, שתחזית המכירות בו משתנה כאשר המשתמש עובר לרשומה אחרת

כדי להוסיף תרשים בדף גישה לנתונים, בחר את האזור הרצוי בדף, ולאחר מכן בחר **הוספה** (Insert), **תרשים של Office** (Office Chart). הפקודה מפעילה אשף שמנחה את המשתמש לאורך תהליך העיצוב של תרשים. תוכל לבחור את סוג התרשים ומקור

הרשומות שלו, ולהגדיר אילו ערכים להציג בו. לאחר שסיימת את השימוש באשף, לחץ לחיצה ימנית בתרשים ובחר באפשרות **ארגז כלים של מאפיינים** (Property Toolbox) כדי לערוך את הפריטים שבחרת או לציין אפשרויות אחרות שלא הוצגו לבחירתך באופן מפורש בעת היצירה.

דוגמת רשימה של טבלת ציר

הפקד של רשימת טבלת ציר מאפשר למשתמש למיין, לקבץ, לסנן, לחלק לרמות ולטפל בנתונים. כמו כן הוא מסוגל לפעול עם נתונים של יותר ספקים בהשוואה לדף גישה לנתונים רגיל, או רכיבי Web אחרים של Office 2000. מקורות הנתונים של פקד זה יכולים להיות גיליון עבודה, מסד נתונים, וקוביית נתונים רב-מימדית.

תרשים 13.17 מציג פקד של רשימת טבלת ציר בדף גישה לנתונים. הדף מבוסס על הטבלה Orders שבמסד הנתונים NorthwindCS. הפקד מונה הזמנות לפי ShipCountry, CustomerID ו- EmployeeID. לחץ על הפקד **הרחב** (Expand) הסמוך לשדות EmployeeID ו- CustomerID כדי להרחיב את הטבלה במסגרת אותה עמודה או שורה. בנוסף, תוכל להציג באופן בררני קבוצת משנה של הטבלה על ידי פתיחת רשימת האלמנטים המכילה את כל המדינות, הלקוחות והעובדים, ובחירה באחד או בכמה מהם מכל קבוצה. תיבת כלים המופיעה בראש רשימת טבלת הציר מכילה עוד אפשרויות ניתוח, כמו למשל סינון ומיון.

Page2

This PivotTable is based on NorthwindCS

dbo.Orders

ShipCountry: All

	EmployeeID					
	⊞ 1	⊞ 2	⊞ 3	⊞ 4	⊞ 5	⊞ 6
CustomerID	Count of OrderID	Count of OrderID	Count of OrderID	Count of OrderID	Count of OrderID	Count of OrderID
⊞ ALFKI	2		1	2		1
⊞ ANATR			2	1		
⊞ ANTON	1		3	1		
⊞ AROUT	3		2	4		1
⊞ BERGS	4	1	6	1	2	
⊞ BLAUS			1	1		1
⊞ BLONP		1	2	3	1	2
⊞ BOLID				2		
⊞ BONAP	3	1	3	4	1	
⊞ BOTTM	2	2	4	2		2
⊞ BSBEV	1	1	2	2		2

תרשים 13.17: רשימת טבלת ציר בדף גישה לנתונים המציג נתונים מתוך מסד הנתונים NorthwindCS

נושאי תכנות הקשורים בדפי גישה לנתונים

בעזרת דפי גישה לנתונים, תוכל לתכנת פתרונות במספר רמות שונות. דפי גישה לנתונים יכולים לשמש גם כמארחים לרכיבי Web של Office. הן דפי גישה לנתונים והן רכיבי Web של Office מכילים תבניות אובייקט שתוכל לנצל לצורך פיתוח פתרונות באמצעות קוד. עוד ניתן לתכנת פתרונות ב- VBA או בשפת scripting ל- Web, כמו למשל VBScript.

סעיף זה בודק פתרונות המבוססים על VBA, על האוסף AllDataAccessPages ועל האובייקט DataAccessPage. האוסף AllDataAccessPages פועל כמו האוספים AllForms ו-AllReports. חבריו אינם אובייקטים של מסד נתונים, כי אם אובייקטים של AccessObject. אובייקטים אלה זמינים בין אם יש דף גישה לנתונים פתוח ובין אם לאו. האוסף מאפשר לך לאתר במלואה את כל קבוצת דפי הגישה לנתונים הקשורה בפרויקט של מסד נתונים. השיגרה הקצרה הבאה ב- VBA רושמת את כל דפי הגישה לנתונים בפרויקט נתון, ומציינת ליד כל אחד אם הוא פתוח או לא.

```
Sub listPages()
Dim myPage As AccessObject
   For Each myPage In Application.CurrentProject.AllDataAccessPages
      Debug.Print myPage.Name & IIf(myPage.IsLoaded, _
         " is loaded.", " is not loaded.")
   Next myPage
End Sub
```

למאפיין FullName באובייקט AccessObject שבאוסף AllDataAccessPages נודעת משמעות מיוחדת. זכור, שדפים אינם נשמרים כאובייקטים בקובץ מסד הנתונים, אלא הם קבצי DHTML נפרדים. מיקומם של הדפים יכול להיות בכל נקודה-שהיא ברשת המקומית. המאפיין FullName מציין את הנתיב ואת שם הקובץ עבור דף גישה לנתונים. השיגרה הבאה רושמת את כל הדפים בפרויקט, ומפרטת אותם לפי המאפיינים Name ו- FullName. ערך המאפיין Name הוא שם מקוצר לדף הגישה לנתונים המופיע בחלון מסד הנתונים.

```
Sub whereArePages()
Dim myPage As AccessObject
Dim obj As Object
   Set obj = Application.CurrentProject
   For Each myPage In obj.AllDataAccessPages
     Debug.Print "The link "; myPage.Name & _
     " points at " & myPage.FullName & "."
   Next myPage
End Sub
```

השיטה היחידה של האובייקט DataAccessPage היא ApplyTheme. השתמש בשיטה זאת כדי ליצור אוטומציה בהחלת ערכת נושא של Office על הדפים שאיברי האוסף AllDataAccessPages מצביעים אליהם. שיטה זאת תפעל באופן תקין רק אם הדף פתוח במצב עיצוב. שתי השגרות הבאות מחילות ערכת נושא על כל דפי הגישה לנתונים בפרויקט, בין אם הדפים פתוחים ובין אם לאו, ובין אם הם פתוחים במצב עיצוב, ובין אם לאו. השגרות גם משחזרות כל דף גישה לנתונים ומחזירות אותו למצב הפתיחה והתצוגה שהיו לו לפני החלת ערכת הנושא. אם אינך מרוצה מערכת הנושא שנבחרה, הפעל מחדש את השיגרה callSetTheme עם הארגומנט Blank.

```
Sub callSetTheme()
' Test with Artsy or Blends.
' Clear with Blank.
   setTheme "Artsy"
End Sub

Sub setTheme(ThemeName As String)
Dim myPage As AccessObject
Dim obj As Object
Dim blnCloseit As Boolean
Dim blnMakePageView As Boolean
'Loop through all DataAccessPages.
   Set obj = Application.CurrentProject
   For Each myPage In obj.AllDataAccessPages

' Get Page open in Design view.
      If myPage.IsLoaded = False Then
         DoCmd.OpenDataAccessPage myPage.Name, acDataAccessPageDesign
         blnCloseit = True
      Else
         If DataAccessPages(myPage.Name).CurrentView <> _
            acDataAccessPageDesign Then
            DoCmd.Close acDataAccessPage, myPage.Name
            DoCmd.OpenDataAccessPage myPage.Name, acDataAccessPageDesign
            blnMakePageView = True
         End If
      End If

' Apply Theme.
      DataAccessPages(myPage.Name).ApplyTheme ThemeName
      DoCmd.Save acDataAccessPage, myPage.Name
```

```
' If necessary, restore page.
        If blnCloseit = True Then
            DoCmd.Close acDataAccessPage, myPage.Name
            blnCloseit = False
        ElseIf blnMakePageView = True Then
            DoCmd.Close acDataAccessPage, myPage.Name
            DoCmd.OpenDataAccessPage myPage.Name, acDataAccessPageBrowse
        End If
    Next myPage
End Sub
```

14

מהדורת Office 2000 למפתחים (ODE)

מהדורת Office 2000 למפתחים (ODE) כוללת רכיבים חשובים רבים לפיתוח יישומים בסביבת Microsoft Access, כולל אפשרויות התקנה והטמעה רב-תכליתיות. מהדורת ODE שווקה במקור עם Office 97 בתור התפתחות טבעית של Access SDK (ערכת משאבי הפיתוח של Access). מפתחים מנוסים ב-Access יכירו בערכו של ODE בתור אמצעי לגישה אל מנגנון זמן ריצה ואשף ההתקנה של Access. ODE עשתה כברת דרך ארוכה מאז, והיא כוללת, בין השאר, רכיבים חדשים או משופרים במידה רבה:

- גרסת זמן ריצה של Microsoft Data Engine (MSDE)
- אשף האריזה וההטמעה
- כלים המאפשרים איגוד נתונים (data binding) ב- Microsoft Forms המשמשים יחד עם יישומי Office נוספים (במקום Access Forms)
- **ספרן הקוד** (Code Librarian) לאחסון ואחזור של תגזירי קוד ושגרות מלאות
- תיעוד מודפס ומקוון בהיקף מלא למפתחי יישומים לסביבת Office 2000

פרק זה מציג סקירה קצרה של קבוצת הרכיבים של ODE ובוחן נושאים בעלי עניין מיוחד למפתחי Access (החתמה דיגיטלית היא נושא שגם מפתחים אחרים ימצאו בו עניין).

סקירה כללית על ODE

היתרונות של ODE באים לידי ביטוי בשלושה תחומים: תפוקת Microsoft Visual Basic for Applications, גישה לנתונים ופיתוח וניהול. ODE מיועדת למיליוני מפתחים מקצועיים שיוצרים פתרונות בעזרת Office, VBA וכלי גישה לנתונים. חלק גדול מפתרונות אלה מנצלים את יכולות הגישה לנתונים רבות העוצמה של Access מתוך יכולות של יישום אחר של Office.

ODE כוללת את מוצרי Microsoft הבאים:

- Microsoft Word 2000
- Microsoft Excel 2000
- Microsoft PowerPoint 2000
- Microsoft Access 2000
- Microsoft Outlook 2000
- Microsoft Publisher 2000
- Microsoft FrontPage 2000
- Microsoft PhotoDraw 2000
- Microsoft Small Business Tools

ODE כוללת תיעוד ודוגמאות שיסייעו לך בבניית פתרונות בתוך יישומים אלה וביניהם. VBA זמין בכל משפחת מוצרי Office, אך הקושי העיקרי בהטמעת השימוש ביישום חדש הוא לימוד מודל האובייקט. התיעוד המודפס של ODE כולל עלון ובו תיאורים גרפיים של מודלי האובייקט של היישומים שנמנו לעיל (למעט PhotoDraw). ODE כוללת תיעוד של 24 מודלי אובייקט המיועדים ליישומים העיקריים, משותפים להם, מותאמים במיוחד לשירותי גישה לנתונים או מיועדים לטכנולוגיות Web שמשובצות ביישומי Office.

כלי VBA לפריון עבודה

כלי VBA לפריון עבודה מסייעים להאיץ את תהליכי כתיבת הקוד בכל יישומי Office שתומכים ב-VBA. מדובר במיגוון כלים רחב, החל בעורך מחרוזות פשוט לבניית משפטי SQL, וכלה במעצב התוספות (Add-In Designer) של COM (Component Object Model), שמפשט את הקמת ספריות הקישור הדינמי (ספריות DLL) בצורת תוספות בשני יישומי Office או יותר.

מעצב התוספות של COM (COM Add-In Designer) מספק גישה משולבת לבנייה ושימוש בתוספות בין כל יישומי Office. גרסאות קודמות של Office חייבו גישות שונות לבנייה, אחסון והפעלה של תוספות.

ספרן הקוד (Code Librarian) הוא מסד נתונים המאפשר לשתף קטעי קוד, פונקציות ומודולים בין חברי צוות הפיתוח. איש פיתוח בודד יכול אף הוא לנצל אותו לאחזור קוד שנכתב בעבר. באפשרותך להוסיף בעצמך קוד מותאם אישית אל מסד הנתונים ולחפש אותו באמצעות מילת מפתח וקריטריונים נוספים. ספרן הקוד מספק ערכת קוד בסיסית עבור מטלות תכנות מקובלות.

פרשן הקוד של VBA (VBA Code Commentator) **ומטפל השגיאות של VBA** (VBA Error Handler) מעדכנים בעצמם את המודולים כדי לבצע משימות רגילות חיוניות בפעילויות פיתוח מותאמות אישית רבות. פרשן הקוד מוסיף הערות, כגון שם איש הפיתוח, התאריך והפרמטרים שאיש הפיתוח צריך לשלוח לשגרות. הוא מרכיב מידע זה מתוך תבנית הניתנת להתאמה אישית. התבנית מאפשרת לצוות פיתוח הפועל בארגון גדול לשנות את התהליך הבסיסי כך שיענה על צורכיהם המיוחדים. מטפל השגיאות בונה קוד מתוקנן (standardized) לטיפול בשגיאות בכל שגרות היישום. ניתן להתאים גם את התבנית שלו באופן אישי. מעבר לכך, מטפל השגיאות מספק סביבת עבודה להוספת קוד טיפול בשגיאות שתכתוב בעצמך.

עורך המחרוזות של VBA (VBA string Editor) הוא תיבת דו-שיח לכתיבה, עריכה, העתקה והדבקה של משפטי SQL מותאמים אשית בקוד VBA. עורך זה הוא כלי פשוט לביצוע משימה יחידה שעלולה להיות מורכבת למדי.

ODE מפשט גם את עבודת הפיתוח בצוות באמצעות Microsoft Visual SourceSafe, שרכיבי המסירה והקבלה שלו מאפשרים לחבר הצוות לטפל במודול VBA אחד או יותר, בעוד שעמיתיו מטפלים בהיבטי אבטחת איכות או ברכיבי שילוב המערכת של שאר חלקי היישום. Visual SourceSafe מאפשר גם להשוות בין גרסאות קוד VBA ולשחזר גרסאות קודמות, כל זאת בצורה קלה ופשוטה.

התוספת **יבוא/יצוא קוד מרובה של VBA** (VBA Multi-Code Import/Export) מאפשרת לייבא קוד אל מודול וגם לייצא אותו ממנו. טוענים רכיב זה בתור תוספת בודדת, אך הוא כולל שני ממשקי משתמש: אחד לייבוא של קוד והאחר לייצוא של קוד.

התיעוד המודפס והמקוון של ODE מאפשר לשפר את מיומנויות הפיתוח ומתאר טכניקות לפיתוח יישומים המשלבות בין רכיבים. נוסף למודולי האובייקט המודפסים, התיעוד כולל את המדריך למתכנת VBA. במצורף לתיעוד המקוון תמצא את האוסף המיוחד של תקליטורי רשת הפיתוח של Microsoft – MSDN (Microsoft Developer Network) ובו כל החומר המודפס ומסמכים נוספים; החומר כולל תמיכה במנגנון חיפוש.

גישה לנתונים

ODE כוללת אוסף כלים התומכים בפיתוח גישה לנתונים. הדובדבן שעל הקצפת הוא, קרוב לוודאי, מנגנון הנתונים של Microsoft המיועד להפצה חוזרת – MSDE (Microsoft Data Engine). באפשרותך לשלוח גירסה עצמאית של יישום מותאם אישית שכתבת, שתומכת ב-MSDE, מבלי לשלם תמלוגים על כך. הדבר מתאים במיוחד לגרסאות

אבטיפוס של יישומים שעשויים להתפתח ליישומי SQL רחבי היקף. חתימת המשאבים של MSDE קטנה בהרבה מזו של SQL Server, ולכן מומלץ להשתמש בו במקרים שדרישת המשאבים של SQL תובענית מדי. ניתן לבנות גם טפסים ודוחות מותאמים אישית כנגד SQL Server בעזרת הרכיב החדש פרויקט Access (עיין בפרק 12 לקבלת מידע נוסף אודות MSDE ופרויקטים של Access).

להבטחת תאימות מירבית לאחור, עליך לבנות פתרונות באמצעות Microsoft Access Run Time. חבילה זו מְדַמָּה את Access 2000 ואת Jet ללא ממשק המשתמש. היישום המותאם אישית שכתבת הוא הממשק החשוף היחיד. חבילה זו, הפטורה מתשלום תמלוגים, מאפשרת לשלוח פתרונות הפועלים ברמת היחידה הארגונית, כאלה שדרישות המשאבים שלהם אינן גדלות במשך הזמן. לדוגמה, ניתן לנצל את Access Run Time כדי ליצור פתרון עבור מפעל המספק שירותי תשתית ציבוריים (כגון מים וחשמל) מפעלי שירותים ציבוריים מתקיימים במשך תקופה ארוכה יחסית ולעיתים קרובות הם מתפקדים ברמה מיטבית או ברמה קרובה לה. דרישות המערכת בסביבה מסוג זה אינן נוטות לגדול משמעותית לאורך חיי היישום.

פקד הנתונים של ADO (ADO Data Control) ומעצב סביבת הנתונים (Data Environment Designer) מאפשרים כלי גישה גרפיים למקורות נתונים ADO ו-ODBC. פקד הנתונים של ADO מוכר למי שהשתמש **במעצב חיבור משתמש ActiveX** (ActiveX User Connection Designer) כדי ליצור אובייקטי נתונים מרוחקים בזמן העיצוב. פקד הנתונים של ADO מאפשר יצירת חיבור למקור נתונים אחד בזמן נתון ובאפשרותו לנווט לרשומה הבאה, הקודמת, האחרונה והראשונה בערכת הרשומות של מקור הנתונים. מעצב סביבת הנתונים מאפשר ליצור אובייקטים רבים מהסוג Connection, שכל אחד מהם כולל אובייקטים רבים מהסוג Command עם ערכות רשומות מתאימות. המעצב מאפשר לגרור ולשחרר שדות מאובייקט מסוג Command ב-Microsoft Forms וגם במעצב דוחות נתונים (Data Report Designer).

כלי גישה גרפיים אלה אינם מבטלים כליל את השימוש בטכניקות התכנות הידניות של ADO. תוכל עדיין להשתמש בקוד של ADO כדי לבנות חיבורי גישה לנתונים אל מקורות נתונים מוכרים ולטפל בהם מתוך יישומי Office אחרים (בהמשך הפרק נציג דוגמה למימוש משימה זו באמצעות Excel).

מעצב דוחות הנתונים ב-VBA ומעצב סביבת הנתונים משמשים יחד לבניית דוחות מותאמים אישית ללא כתיבת קוד כלשהו. ניתן לייצא דוחות שנוצרו בדרך זו בתור מסמכי HTML (Hypertext Markup Language). תוכל ליצור דוחות באמצעים גרפיים, אך השימוש בקוד מעניק לך שליטה גדולה יותר בעיצוב, הדפסה, הצגה לפני הדפסה ושמירה של הדוחות ההירככיים.

הטמעה וניהול

אשף האריזה וההטמעה (Package and Deployment Wizard) של Office 2000 מהווה שיפור משמעותי לעומת אשפי ההתקנה וההפצה בגרסאות קודמות. כל פעולות האשף מונעות באמצעות תפריטים. באפשרותך לנצלו להפקת חבילת התקנה מקצועית

שפועלת בצורה דומה לזו של Office 2000. אחד הרכיבים המלהיבים ביותר שלו היא האפשרות לפרוס פתרונות מותאמים אישית באמצעות האינטרנט, כך שלקוחות מרוחקים יכולים להתקין את הפתרונות המותאמים אישית שתשלח אליהם ללא צורך בדיסקט או בתקליטור. ניתן אף להפוך את פעולות האריזה וההטמעה לאוטומטיות באמצעות scripts מותאמים אישית שמבטלים את האפשרות לטעות בבחירת מסכי האשף.

באפשרותך לנצל את האשף כדי להפיץ פתרונות המבוססים על MSDE או Access Run Time הניתנים לשימוש חוזר. יחד עם זאת, אל לך להתעלם ממתן פתרונות המנצלים את מלוא רכיבי הפעולה של גרסת Access 2000. אחת הסיבות שלקוחות נוטים להשתמש בפתרונות מותאמים אישית רבים היא הכרתם את הסביבה הביצועית של Access. לקוחות אחדים רוצים לקחת חלק בתחזוקת המערכות שלהם, כדי להקטין את עלות הבעלות הכוללת.

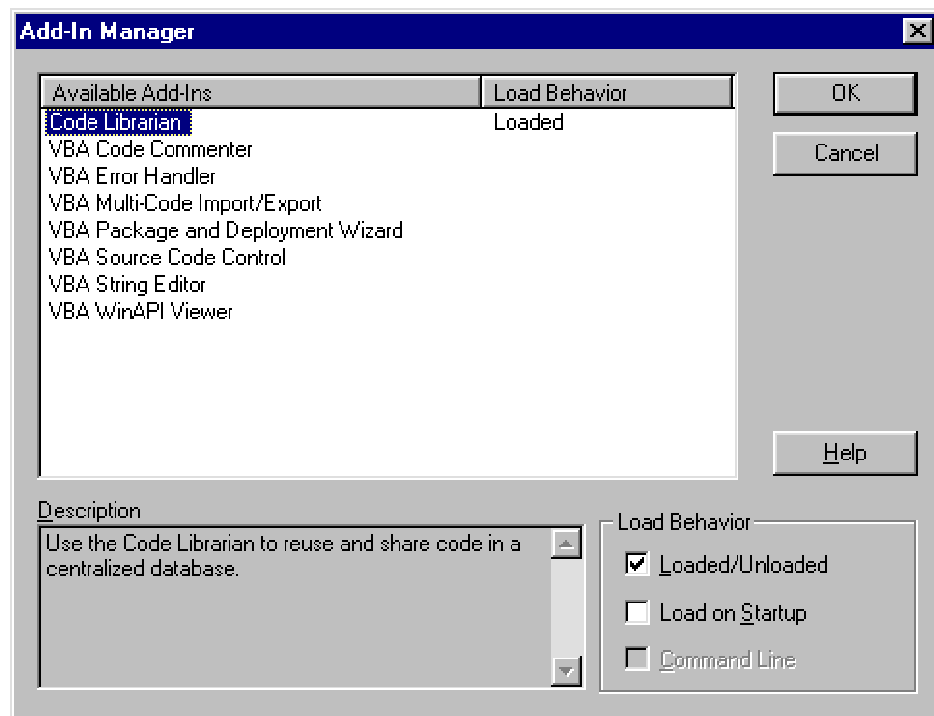
מנהל השכפול של Microsoft (Microsoft Replication Manager), ממשק משתמש גרפי שמשווק עם ODE, מאפשר להציג ולנהל מסדי נתונים Jet משוכפלים ברשת או באינטרנט. מנהל השכפול ממיר מסד נתונים לתבנית בסיס לעיצוב, יוצר עותקים נוספים ומנהל קבוצת עותקים. באפשרותך לנצלו לניהול חילופי נתונים ומבני נתונים בין חברי קבוצת העותקים המשוכפלים, באמצעים גרפיים. מנהל השכפול מאפשר סנכרון בשלושה סגנונות: לפי בקשה, בחיבור הבא ובלוח זמנים רגיל. בנוסף, הוא תומך בחילופי נתונים בין מסדי הנתונים Jet ו- SQL Server (עיין בפרק 11 לקבלת מידע נוסף אודות שכפול מסד נתונים והצגת דוגמאות שממחישות כיצד לנהל שכפול).

כל פתרון מותאם אישית יכול לצאת נשכר ממערכת עזרה או מסמכים כלשהם. משתמשים זקוקים למסמך שמסביר כיצד עליהם להגיב למסכי המערכת, ומפתחים צריכים תרשימי זרימה וקטעי קוד שממחישים את הקשר בין קוד ואובייקטי מסד נתונים כחלק מהתמיכה בביצועי המערכת. ODE משווקת עם HTML Help Workshop, שתומכת בשני התפקודים האלה. כל המסכים המוצגים בפרק זה, כמו מסכים אחרים בשאר חלקי הספר, נלכדו באמצעות הכלי HTML Help Image Editor. כלי זה מאפשר להציג את הסמן (cursor) – רכיב שנעדר מחבילות לכידת מסכים רבות. הוא תומך בלכידת מסך מלא וגם בלכידת חלונות שונים בתוך המסך, בהתאם לקוצב זמן (timer), לחיצה בעכבר או הקשה במקלדת.

ספרן הקוד

כדי לעשות שימוש חוזר בקוד, יש צורך במקום כדי לאחסן אותו, אמצעי לחפש אותו ומנגנון שיאחזר אותו. ספרן הקוד (Code Librarian) המשווק עם ODE מאפשר לעשות את כל הפעולות האלו, ועוד יותר מהן. הפעלת **ספרן הקוד** היא פעולה דו-שלבית (בכך הוא דומה לתוספות מוכללות אחרות של ODE). בפעם הראשונה בהפעלה (session) שתרצה להפעיל את ספרן הקוד, בחר באפשרות **Add-Ins** (תוספות) בחלון עורך VB של פרויקט ולאחר מכן לחץ לחיצה כפולה על **Code Librarian** בתיבת הדו-שיח

Add-In Manager (עיין בתרשים 14.1). המילה **Loaded** (טעון) מופיעה בעמודה **Load Behavior** ונבחרת תיבת הסימון **Loaded/Unloaded** של הקבוצה **Load Behavior**. לחץ לחיצה כפולה על תוספות כלשהן שברצונך לנצל במהלך ההפעלה. באפשרותך לבחור בתיבת הסימון **Load On Startup** כדי לגרום לתוספת להתחיל לפעול מעצמה בעת פתיחת החלון **פרוייקט** (Project). לאחר טעינת התוספת, באפשרותך להפעילה מהתפריט **Add-Ins**.

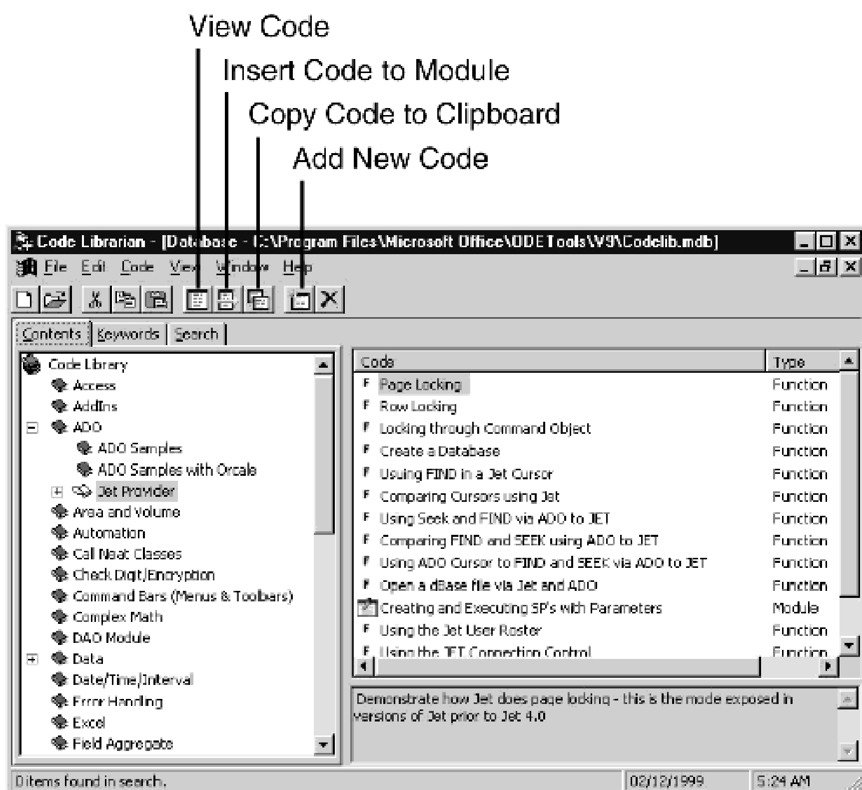


תרשים 14.1: תיבת הדו-שיח **Add-In Manager**

תרשים 14.2 מציג את חלון ספרן הקוד שמארגן את דוגמאות הקוד לפי קטגוריות. ניתן להרחיב את תחום הקטגוריות. באפשרותך לקשר תיאור עם מקטע קוד כדי שתוכל לזהותו. הכרטיסיה **Keywords** (מילות מפתח) מאפשרת לאחזר רשימת דוגמאות קוד הקשורות למילת מפתח. הכרטיסיה **Search** (חיפוש) מאפשרת לאחזר דוגמאות המכילות מילת מפתח הקשורה לדוגמה או המכילות מילת מפתח בטקסט המלא של הקוד.

סרגל הכלים **Code Librarian** כולל כלי ניהול טיפוסיים. לדוגמה, באפשרותך להציג את הקוד שבערך פריט נבחר על ידי לחיצה על לחצן **View Code** (הצג קוד) (ראה תרשים 14.2). לחצן **Insert Code To Module** (הוסף קוד למודול) שמימין מוסיף את הקוד לחלון הקוד במקום הנוכחי שבו נמצא הסמן בחלון, ולכן יש לוודא שהסמן אינו נמצא בשיגרה כלשהי אחרת (אלא אם ברצונך להוסיף את הקטע דווקא אליה). באפשרותך גם לגרור ולשחרר דוגמת קוד מתוך הספרן אל קוד שאתה יוצר. לחצן

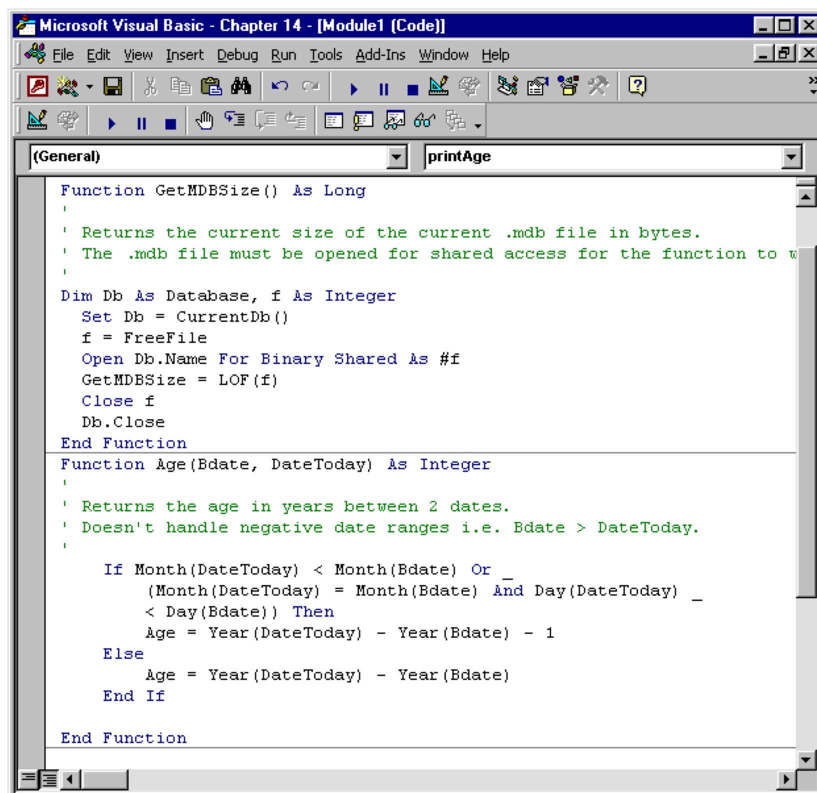
Copy Code To Clipboard (העתק קוד אל הלוח) מאפשר להעתיק קוד ללוח. נוח לעשות זאת כשרוצים להעתיק קוד אל הפעלה (session) אחרת שאינה מוצגת במסך באותו רגע. כל שעליך לעשות הוא לעבור אל הפעלת היעד ולהדביק את תוכן הלוח בחלון הקוד שלה.



תרשים 14.2: Code Librarian מארגן את דוגמאות הקוד לפי קטגוריות, אך ניתן לחפש אותן גם לפי מילת מפתח, או בטקסט המלא

באפשרותך גם לשנות קוד קיים או להוסיף קוד חדש לספרן הקוד. כדי לשנות קוד, פתח את הספרן כדי להציגו, ערוך את השינויים הדרושים ולחץ על **Apply** (החל). סגור את חלון ספרן הקוד כדי להחיל את השינויים. להוספת קוד חדש, לחץ על לחצן סרגל הכלים **Insert New Code** (הוסף קוד חדש); תיפתח תיבת דו-שיח ריקה של ספרן הקוד. הדבק את דוגמת הקוד מהלוח אל תיבת הדו-שיח ותן שם לדוגמה בתיבת הטקסט **Name** (שם). תיבת הטקסט **Name** נמצאת בדיוק מעל למקום בו הדבקת את הקוד. הקצה את הסוג (כגון קטע קוד, פונקציה, מודול או מודול מחלקה) באמצעות הכרטיסיה **Description** (תיאור). הקלד תיאור של הדוגמה בתיבת הטקסט מתחת לתיבת הרשימה הנפתחת **Sample Type** (סוג דוגמה). באפשרותך גם להקצות לדוגמה קטגוריה ומילת מפתח אחת או יותר. לחץ על **OK** כדי לשמור את הדוגמה.

תרשים 14.3 מציג סדרת דוגמאות של ספק Jet של ADO. כפי שניתן לראות, הדוגמאות כוללות שני אוספים נוספים. ספרן הקוד משווק כשהוא כולל אוספים אלה ודוגמאות רבות נוספות בתור ערכה בסיסית. חלק מהדוגמאות עשויות להתאים לשימוש ביישומים המותאמים אישית שתיצור, בתנאי שתערוך בהם שינויים בהיקף כזה או אחר. אחת הדוגמאות שבתרשים מחשבת את גודל הקובץ הנוכחי. הדוגמה השנייה מחשבת את גילו של אדם, בהתבסס על תאריך הולדתו והתאריך הנוכחי.



תרשים 14.3: הדוגמאות המוצגות כלולות ב- Code Librarian

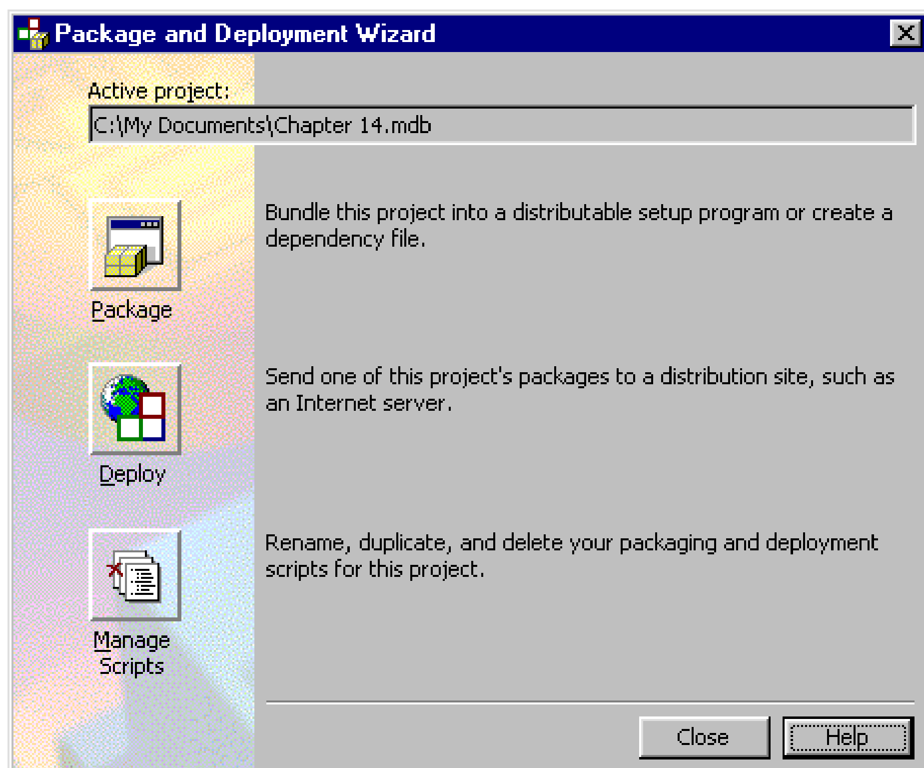
פתרונות אריזה והטמעה

אשף האריזה וההטמעה מבצע שלוש פעולות:

- אריזה – יצירת קובץ דחוס אחד או יותר מסוג cab.* להפצה בדיסקט, בתקליטור, ברשת או באינטרנט.
- הטמעה – ניהול הפצת הקבצים הדחוסים אל אמצעי היעד. קיימים 23 מסכים שתומכים בפעולות האריזה וההטמעה. האשף יוצר בעצמו script שרושם את בחירותיך בעת ביצוע פעולה זו או אחרת.

- ניהול scripts – שמירה, שינוי שם, הסרה ושימוש חוזר ב-scripts. מאחר שמספר המסכים בעלי אפשרויות בחירה רבות יכול להיות גדול מאוד, החזרת ה-scripts יכולה לחסוך זמן ניכר ולצמצם את הסיכוי לשגיאות.

מפעילים את האשף בצורה זהה לזו שמפעילים את ספרן הקוד. טען את התוספת ולאחר מכן בחר אותה מתוך התפריט **Add-Ins** בחלון VBE. בטרם תפעיל את האשף, עליך לטעון את היישום המותאם אישית שברצונך לארוז ולפרוס. תרשים 14.4 מציג את המסך הראשון של האשף.



תרשים 14.4: המסך הראשון של אשף האריזה וההטמעה



הערה:

טעינת התוספת עלולה להימשך שניות אחדות. מד ההתקדמות שיוצג ידווח לך על מהלך הטעינה. אם תנסה להפעיל את התוספת טרם סיום טעינתה, עלולה להתרחש שגיאת זמן ריצה.

באפשרותך להפיץ מחדש פתרונות מותאמים אישית או רכיבים שתיצור באמצעות יישומי Office, למעט אלה שתיצור בעזרת FrontPage ו-OutLook. התמיכה בפרויקט, בשני יישומים אלה, מוגבלת לפרופילי המשתמש ולא לפרויקטים עצמאיים.

באפשרותך להפיץ פרויקטים עצמאיים עבור יישומי Office הנותרים. תוכל גם לכלול קבצים נוספים החיוניים לביצוע פרויקט, כגון תמונות מפת סיביות, קבצי DLL, פקדי ActiveX וקבצי מסמכים אחרים של Office.

בין הפתרונות הפועלים יחד עם מסדי הנתונים Jet או MSDE עשויים להיות Access Run Time או MSDE הניתן להפצה חוזרת, מה שמאפשר לפתרונות המותאמים אישית שתיצור לבצע פונקציות תחזוקה של מסד נתונים במחשבים שלא התקינו בהם את Office 2000. באפשרותך לכלול בקבצי cab.* גם את הקובץ Graph9.exe שתומך בתצוגת הנתונים הגרפית.



הערה:

יישומים המנצלים את **רכיבי האינטרנט של Microsoft Office 2000** (Office 2000 Web Components) מחייבים רשיון תחנת עבודה להפעלת Office 2000. יחד עם זאת, באפשרותך להפיץ בצורה חופשית יישומים העושים שימוש ברכיבים אלה. כמו-כן, לא קיים מנגנון זמן ריצה עבור יישומי Office הרגילים, מלבד זה שנזכר לעיל, ולכן יש להתקין את Excel במחשבים שמיועדים להפעיל פרויקט Excel שמופץ מחדש.

אם תבחר להפיץ את היישום המותאם אישית באמצעות Access Run Time, ייתכן שתרצה לבדוק את היישום באמצעות אפשרות שורת הפקודה runtime/, כדי לדמות את סביבת המחשוב שיראו משתמשים שיפעילו את היישום ללא שימוש ב-Access. כשהטפסים המותאמים אישית מפעילים את Access Run Time, הם חיוניים יותר מתמיד, מכיון שאין למשתמשים גישה אל רכיבי ממשק משתמש מוכלל של Access, כגון חלון מסד נתונים או אפילו תיבות דו-שיח של הודעת שגיאה. מסיבה זו תזדקק לטופס פתיחה המשמש כמסך ניווט (ראה פרק 5) ומנגנון לכידת שגיאות קפדני (ראה פרק 1). ללא מנגנון מעין זה, השגיאות תגרומנה ליישום להסתיים בצורה לא מסודרת, דבר שיבלבל את המשתמשים.

באפשרותך ליצור שני סוגי אריזה בעזרת פונקציית האריזה של האשף. אם בדעתך להפיץ את היישום באמצעות דיסקט, תקליטור או משאב רשת משותף, צור חבילה סטנדרטית. אם ברצונך לפרוס את היישום באמצעות דיסקטים בלבד, בחר באפשרות Multiple Cabs, והאשף יתאים את גודל קבצי cab.* כך שאף אחד מהם לא יהיה גדול יותר מקיבולת של דיסקט רגיל. להפצת היישום באמצעות האינטרנט, עליך ליצור חבילת הפצה לאינטרנט.

גישה לנתונים ביישומים שאינם Access

כזכור, ODE מאפשר גישה לנתונים בשלושה נתיבים: מעצב סביבת הנתונים, פקד נתונים ADO ומודל האובייקט של ADO. ייתכן שבהתחלה תחוש בנוח להשתמש בנתיב מודל האובייקט של ADO (פרק 2 מציג מבוא למודל האובייקט של ADO; פרטים

נוספים ניתן למצוא בפרקים אחרים). הדוגמה שלפניך מאכלסת גיליון עבודה של Excel באובייקטים Connection (חיבור) ו-Recordset (ערכת רשומות) של ADO. הקוד שבדוגמה מעתיק את שמות השדות מהטבלה Products של מסד הנתונים Northwind אל השורה הראשונה של גיליון עבודה של Excel. לאחר מכן, הוא מעתיק ערכי שדות אל השורות הבאות מתוך ערכת הרשומות בהמשך גיליון הנתונים. הדוגמה מופעלת מתוך פרויקט VBA הקשור אל קובץ חוברת עבודה.

```
Sub northwind2XL()
Dim cnn1 As ADODB.Connection
Dim rst1 As ADODB.Recordset
Dim RowCnt, FieldCnt As Integer
' Create instances of the connection and recordset objects.
    Set cnn1 = New ADODB.Connection
    Set rst1 = New ADODB.Recordset
' Set cnn1.ConnectionString to Northwind DSN.
    cnn1.ConnectionString = "DSN=Northwind"
' Open connection and recordset.
    cnn1.Open
    Set rst1.ActiveConnection = cnn1
    rst1.Source = "Select * FROM Products"
    rst1.Open , , adOpenStatic, adLockOptimistic
' Write in column headings in first row.
    RowCnt = 1
    For FieldCnt = 0 To rst1.Fields.Count - 1
        Cells(RowCnt, FieldCnt + 1).Value = rst1.Fields(FieldCnt).Name
        Rows(1).Font.Bold = True
    Next FieldCnt

' Fill rows with records, starting at row 2.
    RowCnt = 2
    While Not rst1.EOF
        For FieldCnt = 0 To rst1.Fields.Count - 1
            Cells(RowCnt, FieldCnt + 1).Value = rst1.Fields(FieldCnt).Value
        Next FieldCnt
        rst1.MoveNext
        RowCnt = RowCnt + 1
    Wend
End Sub
```

מחרוזת החיבור של האובייקט Connection מתבססת על אובייקט DSN. ה-DSN שבדוגמה מצביע אל מסד הנתונים Northwind, אך באפשרותו לציין מסד נתונים כלשהו אחר. השימוש ב-DSN מפשט את עיצוב תחביר מחרוזת החיבור, מכיון שהוא

מאפשר להגדיר את החיבור אל מסד הנתונים בצורה גרפית דרך הסמל ODBC של לוח הבקרה. שים לב גם לתחביר הסטנדרטי המשמש להגדרת חיבור וביסוס ערכת רשומות על החיבור.

השיגרה מיישמת את המאפיין Cells שמפנה באופן משתמע לאובייקט ActiveWorksheet במודל האובייקט של Excel. מציינים תאים בודדים בגיליון העבודה על ידי ציון אינדקס השורה ואחריו אינדקס העמודה. קביעת ערך האינדקס בקטע הקוד המקונן בלולאות מאפשר לקוד לעבור על עמודות לרוחב השורה ועל שורות לאורך הגיליון.

חתימה דיגיטלית של פרויקטי VBA

Microsoft Office 2000/Visual Basic Programmer's Guide הוא מקור עשיר לתיעוד היבטי פעולה קריטיים רבים, כולל חתימה דיגיטלית של פרויקט. אם בידך תעודת Authenticode, תוכל להחתים פרויקטי VBA שתיצור באמצעות Word, Excel, PowerPoint ו-Outlook כך שלא יגרמו להצגת אזהרת המאקרו, גם כשרכיב האבטחה במחשב המשתמש הוגדר לרמת אבטחה גבוהה.

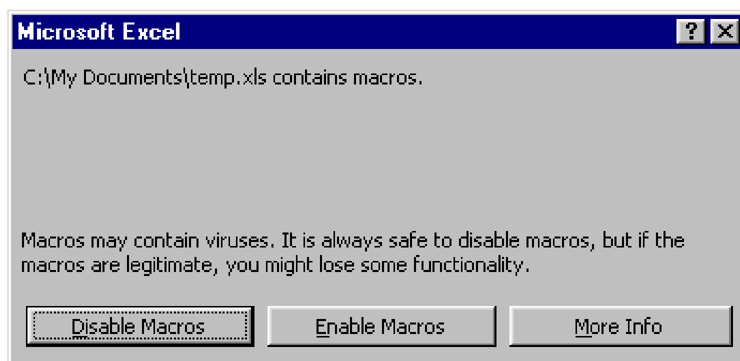
תרשים 14.5 מציג זוג הודעות אזהרה. ההודעה העליונה מוצגת בתגובה למסמכי פרויקט VBA בלתי חתומים. ההודעה התחתונה מוצגת כשמנסים לפתוח מסמכים המכילים פרויקטים חתומים שעדיין לא הוגדרו כאמינים. לא ניתן להוסיף חתימה דיגיטלית ישירות לרשימת המקורות האמינים – עליך לבחור תחילה את תיבת הסימון **Always trust macros from this source** (בטח תמיד במאקרו ממקור זה) ולהפוך את המאקרו שלה לזמינים. כתוצאה מכך, ניתן יהיה לבטוח בכל המסמכים הנושאים את החתימה הדיגיטלית של מסמך זה, ולא תוצגנה אזהרות מאקרו כלשהן בפתיחת פרויקטי VBA שהגיעו מהמקור הנדון.

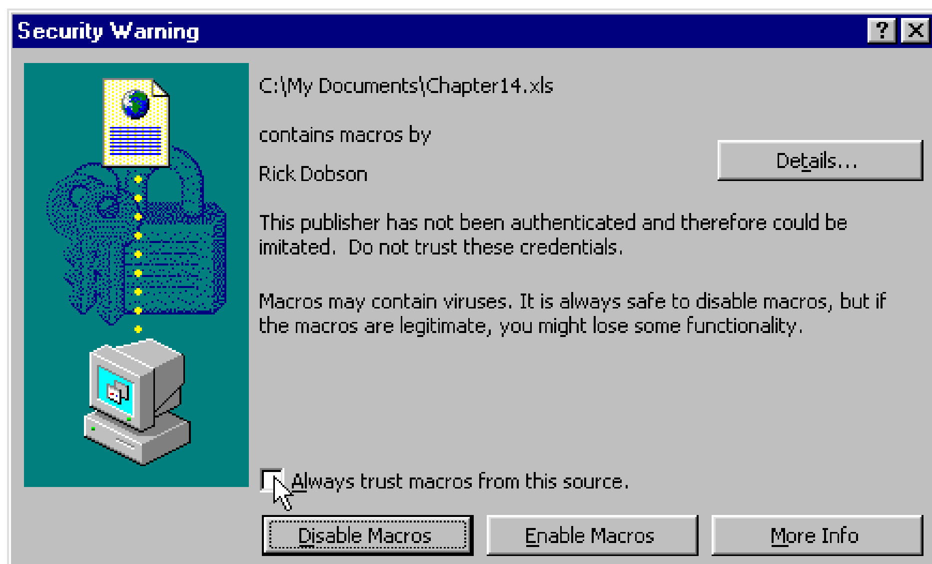


הערה:

אימות באמצעות Authenticode פועל רק בתחנות עבודה שמפעילות את Internet Explorer בגירסה 4 ואילך. לא ניתן להתקין את תוכנת האימות Authenticode בתחנות עבודה שבהן פועלת גירסה מוקדמת יותר של הדפדפן Internet Explorer, או שמותקן בהן דפדפן מסוג אחר.

טכנולוגיית Authenticode מתבססת על הצפנה באמצעות מפתח ציבורי להחתמת פרסומי תוכנה, כגון פרויקטי VBA. כשמפתח תוכנה שומר קובץ, תוכנת Authenticode מבצעת פעולת hashing ליצירת "טביעת אצבע דיגיטלית" במסמך ומצפינה את טביעת האצבע באמצעות מפתח פרטי. כאשר הנמען מקבל את המסמך, תוכנת Authenticode מנסה לשחרר את טביעת האצבע ולפענח אותה. ניתן לפתוח מסמכים שעוברים תהליך זה ללא הצגת הודעות האזהרה. מסמכים שלא עברו את התהליך, יציגו את ההודעה. אם משתמש כלשהו אחר עורך שינויים בפרויקט VBA ושומר אותו, החתימה הדיגיטלית מסולקת מהמסמך, ולכן פתיחתו בפעם הבאה תגרום להצגת הודעת אזהרה.





תרשים 14.5: הודעת אזהרה טיפוסית מפני מאקרו, שמוצגת בפתיחת מסמך בלתי חתום, והודעת אזהרה במסמך ראשון שהגיע ממקור חדש חתום דיגיטלית

להחתמה דיגיטלית של פרויקט VBA עליך להשיג תחילה אישור דיגיטלי חוקי. ניתן לעשות זאת בשלוש דרכים: הדרך הראשונה היא ליצור בעצמך חתימה באמצעות תוכנית השירות Selfcert.exe הכלולה ב- Office 2000. דרך זו מתאימה למטרות בדיקות אישיות או לקבוצת עבודה. הדרך השנייה היא לרכוש אישור באמצעות רשות האישורים הפנימית של הארגון. דרך זו מתאימה להפצת פרויקטים בארגון או בארגון גדול ובקרב ספקיו. הדרך השלישית היא השגת אישור דיגיטלי מרשות אישורים מסחרית. זהו הפתרון הכללי הטוב ביותר.

החתימה הדיגיטלית ישימה לפרויקט VBA בלבד, ולא למסמכי Office הקשורים אליו, כגון חוברת עבודה. הדבר מאפשר למשתמש לשנות את המסמך הבודד (חוברת עבודה), אך לא את פרויקט VBA שאליו היא שייכת. בחלון VBE, בחר את הפרויקט

בחלון **Project Explorer** (סייר הפרויקט). לאחר מכן בחר **Digital Signature** (חתימה דיגיטלית) מתפריט **Tools** (כלים). בפעם הראשונה שתשתמש באישור, בחר אותו ולחץ פעמיים על **OK**. אחרת, לחץ על **OK** בלבד.

זוג השגרות שלפניך מדגים כיצד ניתן לקבוע אם קובץ חוברת עבודה הוחתם דיגיטלית. השיגרה callIsDSigned קוראת לפונקציה isDSigned פעמיים. בקריאה הראשונה, callIsDSigned מעבירה את שם הקובץ הראשון. בקריאה השנייה, callIsDSigned מציגה דרך אפשרית להשגת שם הקובץ מהמשתמש והעברתו אל isDSigned. חוברת העבודה צריכה להיות פתוחה כדי שניתן יהיה לקבוע אם הוחתמה דיגיטלית, ולכן callIsDSigned פותחת את הקובץ בשם שסיפק המשתמש.

```
Sub callIsDSigned()
Dim myName As String
' Check whether current workbook is digitally signed.
   myName = Application.Workbooks(1).Name
   isDSigned (myName)
' Prompt user for workbook name and
' check whether that workbook is digitally signed.
   myName = InputBox("Type name as filename.xls", _
      "Programming Microsoft Access 2000", myName)
   Workbooks.Open myName
   isDSigned (myName)
End Sub

Public Function isDSigned(fileName As String)
   isDSigned = Application.Workbooks(fileName).VBASigned
   If isDSigned Then
      Debug.Print fileName & " is digitally signed."
   Else
      Debug.Print fileName & " is not digitally signed."
   End If
End Function
```

הפונקציה isDSigned מאחסנת את ערך ההחזרה של המאפיין VBASigned ולאחר מכן מסתעפת למשפט אחד מתוך זוג משפטי הדפסה בהתאם לערך המוחזר. אם חוברת העבודה מוחתמת דיגיטלית, המאפיין מקבל ערך True. באפשרותך לנצל קוד דומה לזה כדי לקבוע אם מסמכי Word או מצגות PowerPoint הוחתמו דיגיטלית (נוסח התחביר המדויק לפתיחת הקובץ משתנה מיישום ליישום, אך Word ו-PowerPoint תומכים אף הם במאפיין VBASigned).

אינדקס

האינדקס הינו בשפה אנגלית.

לשם נוחות הקריאה התחלנו אותו מסוף הספר. כלומר, סופו בעמוד הבא ותחילתו כמו בספר באנגלית.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z

הוצאת הוד-עמי ספרי מחשבים

ת.ד. 6108 הרצליה 46160 **הזמנות באתר**
info@hod-ami.co.il www.**hod-ami**.co.il

קטלוג ינואר 2017

	כולל	עמ'	מחיר*
אינטרנט - מפתחי אתרים/גרפיקה			
אמא, אבא - בניתי אתר באינטרנט (HTML)		256	29
הגדל את הכנסות העסק שלך באמצעות **פרסום בגוגל Google AdWords**		300	249
HTML5 המדריך לבניית אתרים ולמערכות WEB, הדור הבא – מהד' 2		390	159
The **Java** Tutorial סדנת לימוד	CD	768	179
JavaScript סדנת לימוד		586	159
מדריך **ASP.NET MVC 4**		514	199
ASP.NET 3.5 סדנת לימוד בשפות #C ו-VB		824	99
תכנות			
Code Complete – מדריך מעשי לפיתוח תוכנה		288	139
לחפש באגים, מדריך מעשי לבודק תוכנה, מהד' 3		350	169
Visual C# 3.0 סדנת לימוד		656	99
ללמוד **C** - מהד' 3		480	139
שפת **אסמבלי** למחשב האישי, מהד' 2		314	89
יסודות התכנות ב-**VBA** לתוכנת **Excel**, מהד' 4		152	95
PC - חומרה, תוכנה ורשתות			
מדריך Hacking ואבטחת מידע, מהד' 2		428	169
מדריך **חומרה ותוכנה** לטכנאי PC - מהד' 5 (כולל חלונות 7/8)		752	189
מדריך **רשתות** לטכנאי PC ולמנהלי רשת - מהד' 4		608	219
Windows			
Windows 8.1 מדריך למשתמש		544	149
Windows 8 מדריך למשתמש		438	19
Windows 7 צעד-אחר-צעד		272	19
LINUX			
LINUX למתקדמים, טיפים, טריקים ותכנות ב-BASH חדש		226	189

* מחיר מומלץ לצרכן כולל מע"מ

היכנס לאתר להתעדכן בספרים החדשים ובמחירי המבצע בהוצאה

תוכן עניינים ופרקים לדוגמה www.**hod-ami**.co.il

	כולל	עמ'	מחיר*
גרפיקה			
Flash – ספר הדרכה ותרגילים		122	64
אינדיזיין – ספר הדרכה ותרגילים		132	72
Illusatrator – ספר הדרכה ותרגילים		120	64
Photoshop צעד אחר צעד (צבע מלא, למתחילים), מהד' 3 כריכה קשה		200	159
Photoshop צעד אחר צעד (ש/ל, למתחילים), מהד' 3		200	89
מדריך לתוכנת העיצוב והאנימציה **3ds max** (2 כרכים)	CD	1400	289
OFFICE			
יישומי סטטיסטיקה בגיליון אלקטרוני Excel		86	69
סטטיסטיקה יישומית		150	97
טבלאות ציר – ניתוח נתונים חכם		116	87
יסודות התכנות ב-**VBA** לתוכנת **Excel**, מהד' 4		152	95
Access 2016 צעד אחר צעד חדש		384	169
Word 2016 צעד אחר צעד		150	59
Excel 2016 צעד אחר צעד		336	129
PowerPoint 2010 צעד אחר צעד		202	69
Outlook 2010 צעד אחר צעד		190	69
Access 2010 סדנת לימוד		360	179
עוד ספרים בגרסאות קודמות (2007 ו-2003) ניתן למצוא באתר הוד-עמי			
ניהול, כלכלה ושונות			
לחפש באגים, מדריך מעשי לבודק תוכנה, מהד' 3		350	169
ניהול ממוקד לעשות יותר עם מה שיש (כריכה קשה) - מהד' 4		358	133
לי זה עולה יותר (תמחיר) (כריכה קשה) - מהד' 3		368	129
מערכות מידע			
Oracle SQL יכולות מתקדמות		626	249
SAS (**Statistical Analysis System**) – ספר לימוד		256	169
בסיסי נתונים ושפת **SQL** – עקרונות ועיצוב		648	149
ניתוח מערכות מידע כולל מתודולוגיית ה- **UML**		818	229
המדריך העברי השלם **UML**		346	329
ספרים דיגיטליים			
לרכישת ספרים לצפייה בפורמט PDF היכנס לאתר לקטגוריה "ספרים דיגיטליים"			
קבצי תרגול לספרים			
קבצי תרגול לספרים שונים תמצא באתר בקטגוריה "קבצי תרגול לספרים"			

* מחיר מומלץ לצרכן כולל מע"מ. קטלוג 1/2017

היכנס לאתר להתעדכן בספרים החדשים ובמחירי המבצע בהוצאה

תוכן עניינים ופרקים לדוגמה www.**hod-ami**.co.il